WEEK 6

vl dharshini
231501038
aiml a

1) Determine the factors of a number (i.e., all positive integer values that evenly divide into a number) and then return the p^{th} element of the list, sorted ascending. If there is no p^{th} element, return 0.

Example

n = 20

p = 3

The factors of 20 in ascending order are {1, 2, 4, 5, 10, 20}. Using 1-based indexing, if p = 3, then 4 is returned. If p > 6, 0 would be returned.

Constraints

$1 \leq n \leq 10^{15}$

$1 \leq p \leq 10^{9}$

The first line contains an integer n, the number to factor.

The second line contains an integer p, the 1-based index of the factor to return.

Sample Case 0

Sample Input 0

10

3

Sample Output 0

5

Explanation 0

Factoring n = 10 results in {1, 2, 5, 10}. Return the p = 3^{rd} factor, 5, as the answer.

Sample Case 1

Sample Input 1

10

5

Sample Output 1

0

Explanation 1

Factoring n = 10 results in {1, 2, 5, 10}. There are only 4 factors and p = 5, therefore 0 is returned as the answer.

Sample Case 2

Sample Input 2

1

1

Sample Output 2

1

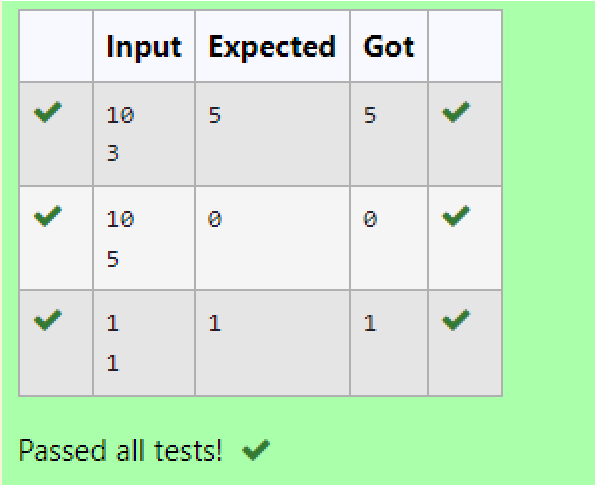
Explanation 2

Factoring n = 1 results in {1}. The p = 1st factor of 1 is returned as the answer.

PROGRAM:

```
a=int(input())
b=int(input())
c=[]
for i in range(1,a+1):
    if a%i==0:
        c.append(i)
if b<=len(c):
    print(c[b-1])
else:
    print("0")
```

OUTPUT:



	Input	Expected	Got	
✓	10 3	5	5	✓
✓	10 5	0	0	✓
✓	1 1	1	1	✓

Passed all tests! ✓

2) Given an array A of sorted integers and another non negative integer k, find if there exists 2 indices i and j such that A[i] - A[j] = k, i != j.

Input Format

1. First line is number of test cases T. Following T lines contain:
2. N, followed by N integers of the array
3. The non-negative integer k

Output format

Print 1 if such a pair exists and 0 if it doesn't.

Example

Input

1

3

1

3

5

4

Output:

1

Input

1

3

1

3

5

99

Output

0

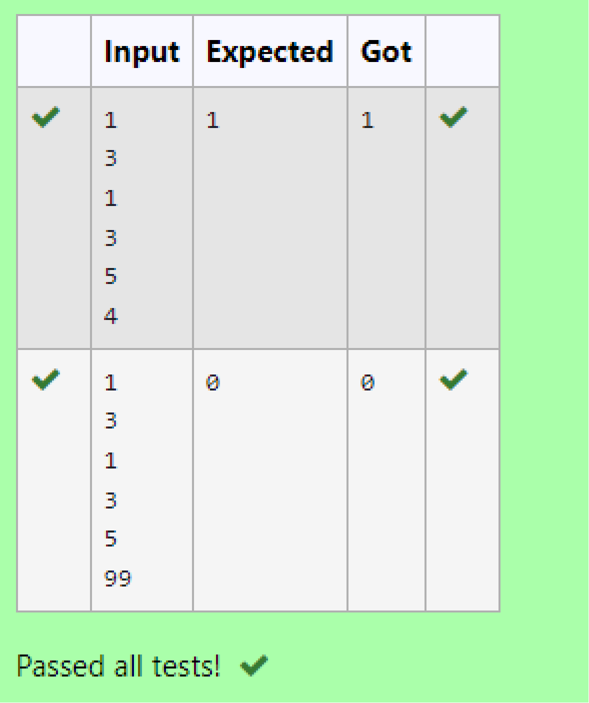
PROGRAM:

```
t=int(input())

for i in range(0,t):
```

WEEK 6

```
n=int(input())
l=[]
for j in range(0,n):
    a=int(input())
    l.append(a)
p=int(input())
for k in range(0,n):
    c=0
    for m in range(i+1,n):
        if l[m]-l[k]==p:
            c=1
            print('1')
            break
    if c==1:
        break
if c==0:
    print('0')
```

OUTPUT:



	Input	Expected	Got	
✓	1 3 1 3 5 4	1	1	✓
✓	1 3 1 3 5 99	0	0	✓

Passed all tests! ✓

3) Write a Python program to check if a given list is strictly increasing or not. Moreover, If removing only one element from the list results in a strictly increasing list, we still consider the list true

Input:

n : Number of elements

List1: List of values

Output

Print "True" if list is strictly increasing or decreasing else print "False"

Sample Test Case

Input

7

1

2

3

0

4

5

6

Output

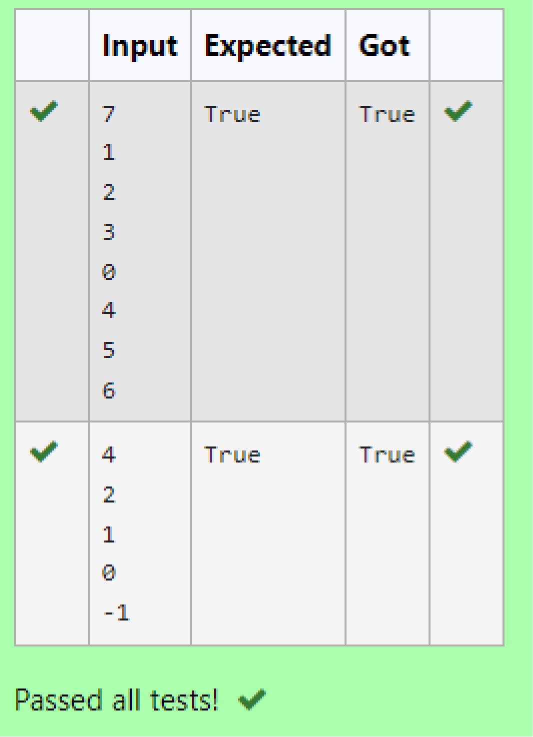
```
True
```

PROGRAM:

```
n = int(input(""))
list1 = [int(input()) for _ in range(n)]

def is_strictly_increasing(lst):
    count = 0
    for i in range(1, len(lst)):
        if lst[i] <= lst[i - 1]:
            count += 1
            if count > 1:
                return False
            # Check if removing the current or previous element helps
            if i == 1 or lst[i] > lst[i - 2]:
                continue
            elif i < len(lst) - 1 and lst[i + 1] > lst[i - 1]:
                continue
            else:
                return False
    return True

def is_strictly_decreasing(lst):
    reversed_lst = lst[::-1]
    return is_strictly_increasing(reversed_lst)

if is_strictly_increasing(list1) or is_strictly_decreasing(list1):
    print("True")
```

```
else:
   print("False")
```

OUTPUT:



	Input	Expected	Got	
✓	7 1 2 3 0 4 5 6	True	True	✓
✓	4 2 1 0 -1	True	True	✓

Passed all tests! ✓

4) Write a Python program to Zip two given lists of lists.

Input:

m : row size

n: column size

list1 and list 2 : Two lists

Output

Zipped List : List which combined both list1 and list2

Sample test case

Sample input

2

2
1

3

```
5
7
2
4
6
8
```

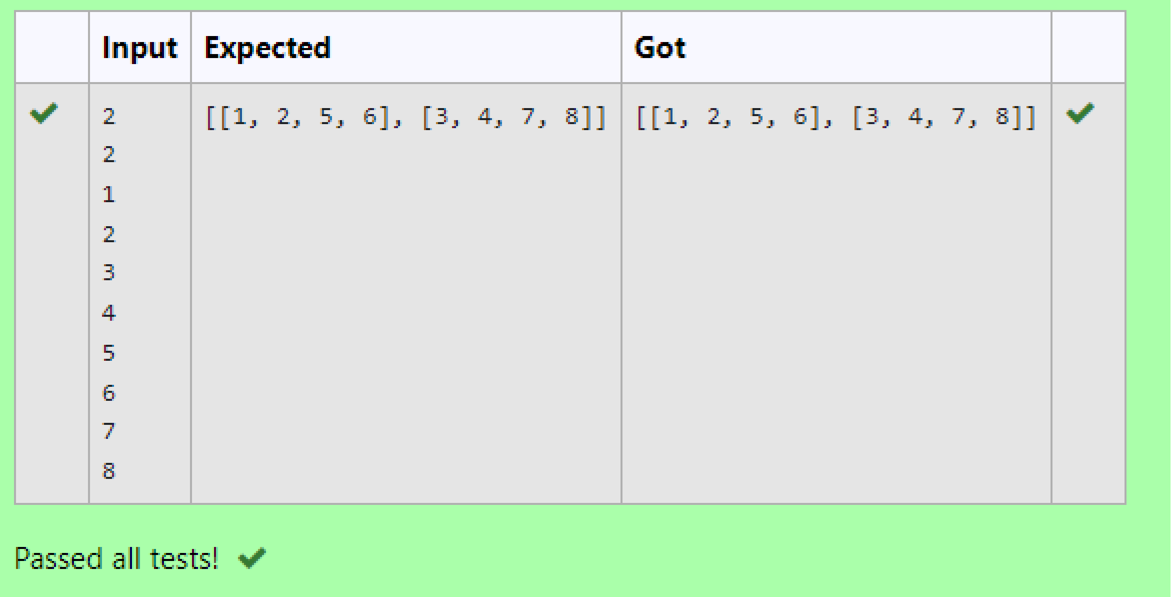
Sample Output

```
[[1, 3, 2, 4], [5, 7, 6, 8]]
```

PROGRAM:

```
l=[]
l1=[]
l2=[]
m=int(input())
n=int(input())
for j in range(m):
    for i in range(n):
        e1=int(input())
        l1.append(e1)
    for i in range(n):
        e2=int(input())
        l2.append(e2)
l.append(l1)
l.append(l2)
print(l)
```

OUTPUT:



	Input	Expected	Got	
✓	2 2 1 2 3 4 5 6 7 8	[[1, 2, 5, 6], [3, 4, 7, 8]]	[[1, 2, 5, 6], [3, 4, 7, 8]]	✓

Passed all tests! ✓

5) Given two lists A and B, and B is an anagram of A. B is an anagram of A means B is made by randomizing the order of the elements in A.

We want to find an *index mapping* P, from A to B. A mapping P[i] = j means the ith element in A appears in B at index j.

These lists A and B may contain duplicates. If there are multiple answers, output any of them.

For example, given

Input

5

12 28 46 32 50

50 12 32 46 28

Output

1 4 3 2 0

Explanation

A = [12, 28, 46, 32, 50]

B = [50, 12, 32, 46, 28]

We should return

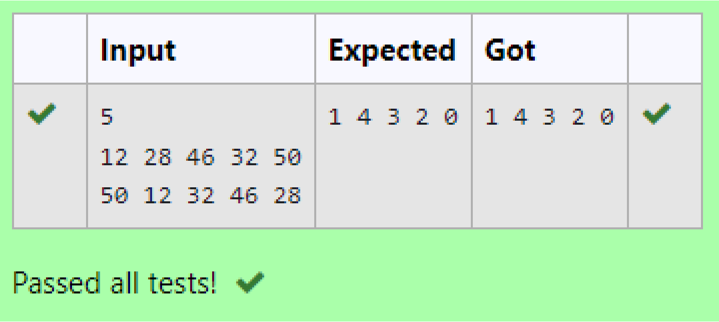
[1, 4, 3, 2, 0]

as P[0] = 1 because the 0th element of A appears at B[1], and P[1] = 4 because the 1st element of A appears at B[4], and so on.

PROGRAM:

```
k = int(input())

v = []

q = list(map(int,input().split()))

d = list(map(int,input().split()))

for i in q:

   v.append(str(d.index(i)))

print(" ".join(v))
```

OUTPUT:



	Input	Expected	Got	
✓	5 12 28 46 32 50 50 12 32 46 28	1 4 3 2 0	1 4 3 2 0	✓

Passed all tests! ✓

6) Program to print all the distinct elements in an array. Distinct elements are nothing but the unique (non-duplicate) elements present in the given array.

Input Format:

First line take an Integer input from stdin which is array length n.

Second line take n Integers which is inputs of array.

Output Format:

Print the Distinct Elements in Array in single line which is space Separated

Example Input:

5

1

2

2

3

4

Output:

1 2 3 4

Example Input:

6

1

1

2

2

3

3

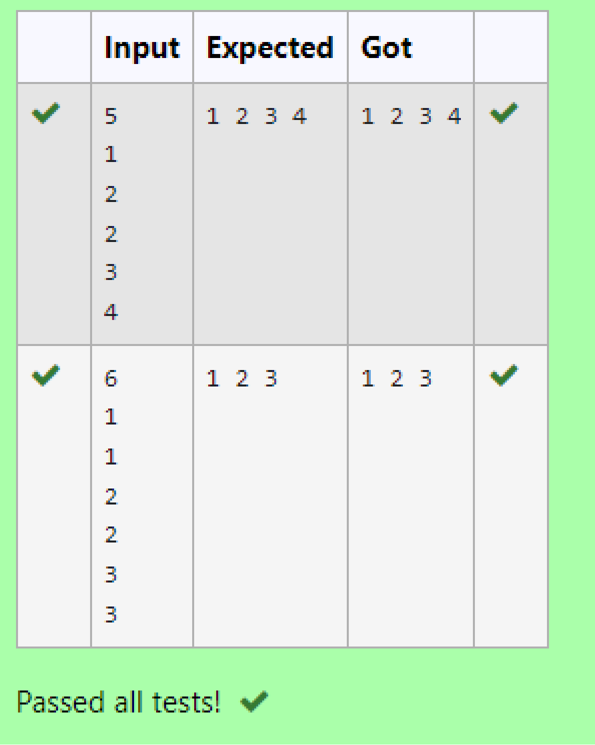
Output:

1 2 3

PROGRAM:

```
a=int(input())
l=[]
for i in range (a):
   b=int(input())
   l.append(b)
print(*set(l))
```

OUTPUT:



	Input	Expected	Got	
✓	5 1 2 2 3 4	1 2 3 4	1 2 3 4	✓
✓	6 1 1 2 2 3 3	1 2 3	1 2 3	✓

Passed all tests! ✓

7) Find the intersection of two sorted arrays.

OR in other words,

Given 2 sorted arrays, find all the elements which occur in both the arrays.

Input Format

The first line contains T, the number of test cases. Following T lines contain:

1. Line 1 contains N1, followed by N1 integers of the first array
2. Line 2 contains N2, followed by N2 integers of the second array

Output Format

The intersection of the arrays in a single line

Example

Input:

1

3 10 17 57

6 2 7 10 15 57 246

Output:

10 57

Input:

```
1
7
1
2
3
3
4
5
6
2
1
6
```

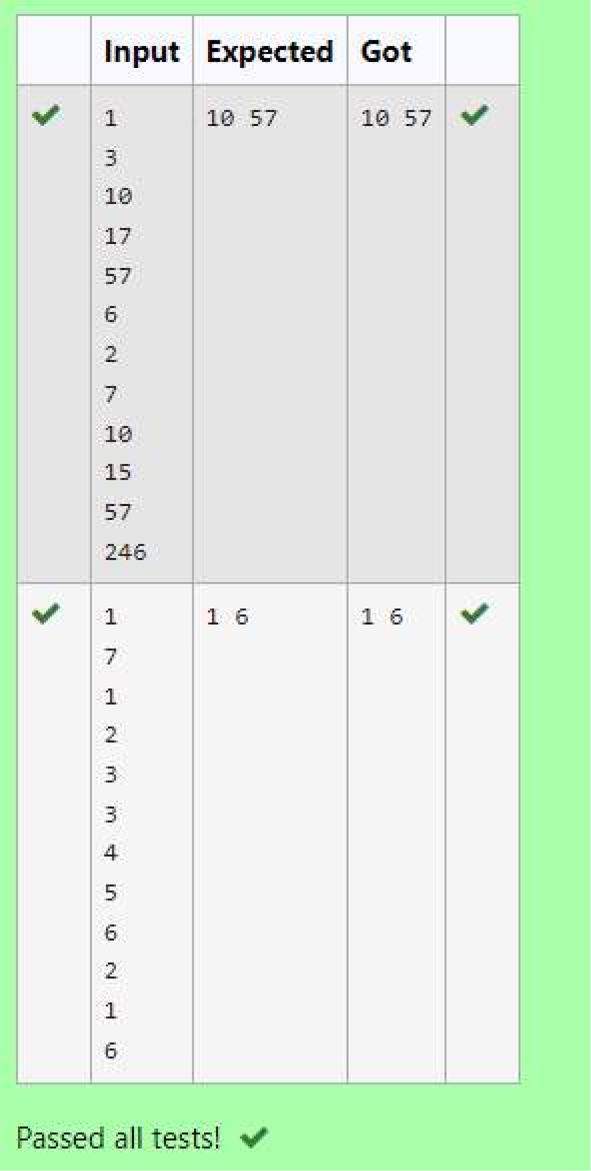
Output:

```
1 6
```

PROGRAM:

```
def intersection(l1,l2):
    l3=[value for value in l1 if value in l2]
    return l3
n=int(input())
for i in range(0,n):
    s1=int(input())
    l1=[]
    for x in range(0,s1):
        e1=int(input())
        l1.append(e1)
    s2=int(input())
    l2=[]
    for y in range(0,s2):
        e2=int(input())
        l2.append(e2)
    print(*intersection(l1,l2))
```

OUTPUT:



	Input	Expected	Got	
✓	1 3 10 17 57 6 2 7 10 15 57 246	10 57	10 57	✓
✓	1 7 1 2 3 3 4 5 6 2 1 6	1 6	1 6	✓

Passed all tests! ✓

8) Consider a program to insert an element / item in the sorted array. Complete the logic by filling up required code in editable section. Consider an array of size 10. The eleventh item is the data is to be inserted.

Sample Test Cases

Test Case 1

Input

```
1
3
4
5
6
7
8
9
10
11
2
```

Output

```
ITEM to be inserted:2
After insertion array is:
1
2
3
4
5
6
7
8
9
10
11
```

Test Case 2

Input

```
11
22
33
55
66
77
88
99
110
120
44
```

Output

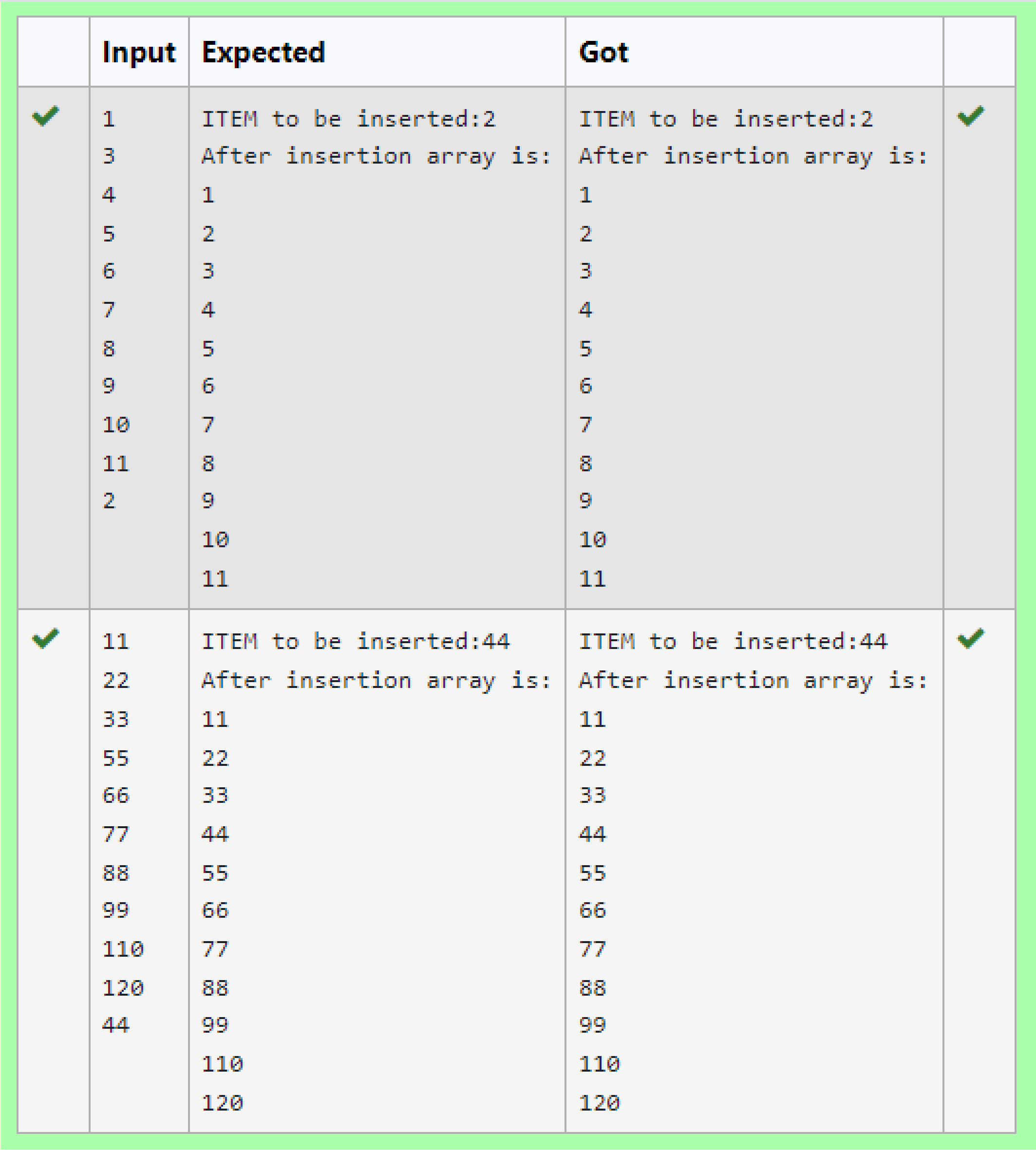
```
ITEM to be inserted:44
After insertion array is:
11
22
33
44
55
66
77
88
99
110
120
```

PROGRAM:

```
l=[]
for i in range(0,10):
    e=int(input())
    l.append(e)
a=int(input())
print("ITEM to be inserted:{:d}".format(a))
print("After insertion array is:")
```

```
l.append(a)

l.sort()

for j in range(0,11):

   print(l[j])
```

PROGRAM:



	Input	Expected	Got	
✓	1 3 4 5 6 7 8 9 10 11 2	ITEM to be inserted:2 After insertion array is: 1 2 3 4 5 6 7 8 9 10 11	ITEM to be inserted:2 After insertion array is: 1 2 3 4 5 6 7 8 9 10 11	✓
✓	11 22 33 55 66 77 88 99 110 120 44	ITEM to be inserted:44 After insertion array is: 11 22 33 44 55 66 77 88 99 110 120	ITEM to be inserted:44 After insertion array is: 11 22 33 44 55 66 77 88 99 110 120	✓

9) Output is a merged array without duplicates.

Input Format

N1 - no of elements in array 1

Array elements for array 1

N2 - no of elements in array 2

Array elements for array2

Output Format

Display the merged array

Sample Input 1

```
5
1
2
3
6
9
4
2
4
5
10
```

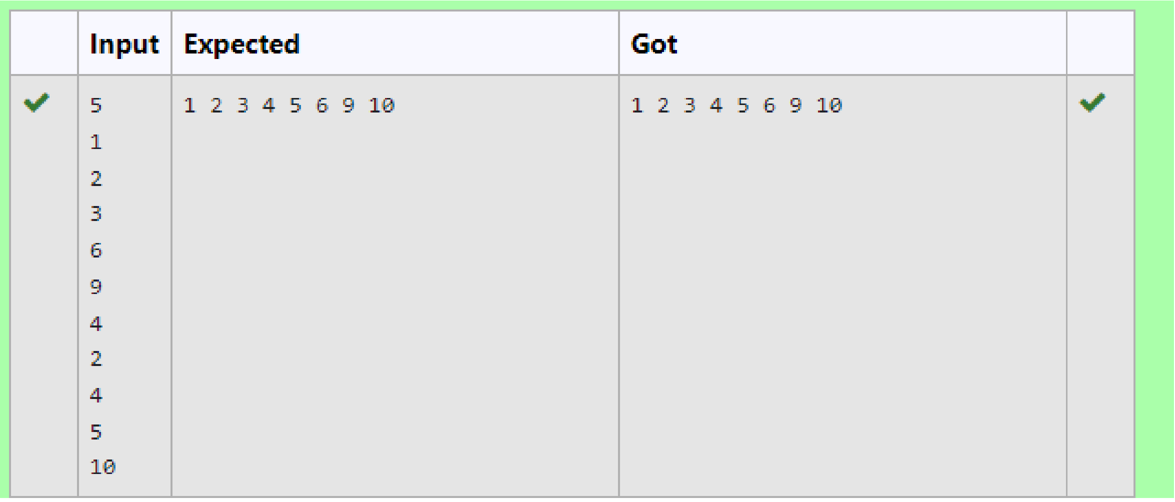
Sample Output 1

```
1 2 3 4 5 6 9 10
```

PROGRAM:

```
a=[]
b=[]
c=int(input())
for i in range(c):
   d=int(input())
   a.append(d)
e=int(input())
for j in range(e):
    f=int(input())
    b.append(f)
for k in range(len(b)):
    if b[k] not in a:
        a.append(b[k])
a.sort()
for l in range(len(a)):
    print(a[l],end=" ")
```

OUTPUT:



	Input	Expected	Got	
✓	5 1 2 3 6 9 4 2 4 5 10	1 2 3 4 5 6 9 10	1 2 3 4 5 6 9 10	✓

10) Write a program to print all the locations at which a particular element (taken as input) is found in a list and also print the total number of times it occurs in the list. The location starts from 1.

For example, if there are 4 elements in the array:

5
6
5
7

If the element to search is 5 then the output will be:

```
5 is present at location 1
5 is present at location 3
5 is present 2 times in the array.
```

Sample Test Cases

Test Case 1

Input

```
4
5
6
5
7
5
```

Output

```
5 is present at location 1.
5 is present at location 3.
5 is present 2 times in the array.
```

Test Case 2

Input

```
5
67
80
45
97
100
50
```

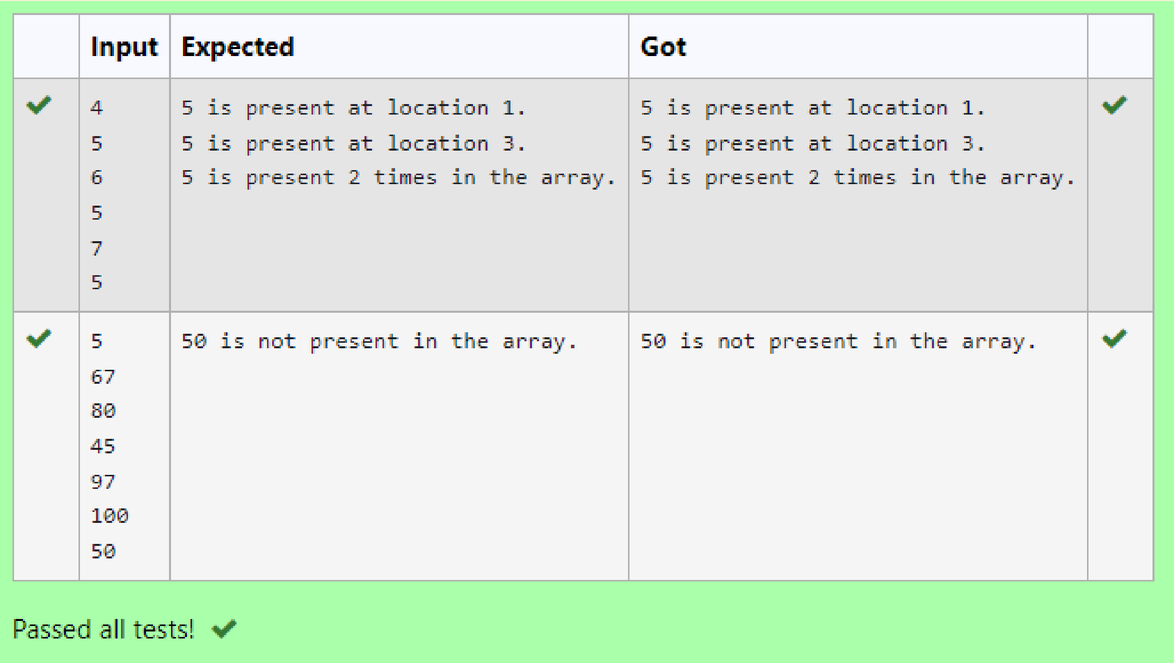
Output

```
50 is not present in the array.
```

PROGRAM:

```
A=[]
n=0
a=int(input())
for i in range(0,a):
  b=int(input())
  A.append(b)
se=int(input())
for j in range(a):
  if(se==A[j]):
    print(se,"is present at location {:d}.".format(j+1))
    n=n+1
if(n==0):
   print(se,"is not present in the array.")
else:
  print(se,"is present {:d} times in the array.".format(n))
```

OUTPUT:



	Input	Expected	Got	
✓	4 5 6 5 7 5	5 is present at location 1. 5 is present at location 3. 5 is present 2 times in the array.	5 is present at location 1. 5 is present at location 3. 5 is present 2 times in the array.	✓
✓	5 67 80 45 97 100 50	50 is not present in the array.	50 is not present in the array.	✓

Passed all tests! ✓