### **[05-List in Python](https://www.rajalakshmicolleges.net/moodle/course/view.php?id=84#section-5)**

**Ex. No. : 5.1 Date:**

**Register No.: 231801024 Name: Boopesh.R**

**Balanced Array**

Given an array of numbers, find the index of the smallest array element (the pivot), for which the sums of all elements to the left and to the right are equal. The array may not be reordered.

Example

arr=[1,2,3,4,6]

·         the sum of the first three elements, 1+2+3=6. The value of the last element is 6.

·         Using zero based indexing, arr[3]=4 is the pivot between the two subarrays.

·         The index of the pivot is 3.

Constraints

·         3 ≤ n ≤ 105

·         1 ≤ arr[i] ≤ 2 × 104, where 0 ≤ i< n

·         It is guaranteed that a solution always exists.

The first line contains an integer n, the size of the array arr.

Each of the next n lines contains an integer, arr[i], where 0 ≤ i< n.

Input

1

3

1

3

5

4

Output:

1

Input

1

3

1

3

5

99

Output

0

**For example:**

| **Input** | **Result** |
| --- | --- |
| 1  3  1  3  5  4 | 1 |
| 1  3  1  3  5  99 | 0 |

**OUTPUT:**

n = int(input())

arr = []

for \_ in range(n):

arr.append(int(input()))

total\_sum = sum(arr)

left\_sum = 0

for i in range(len(arr)):

total\_sum -= arr[i]

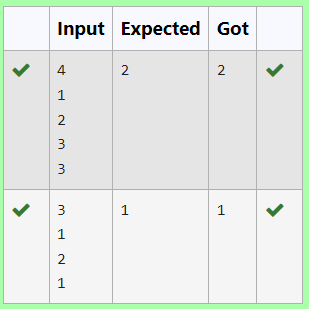
if left\_sum == total\_sum:

print(i)

break

left\_sum += arr[i]

**OUTPUT:**

****

**Ex. No. : 5.2 Date:**

**Register No.: 231801024 Name: Boopesh.R**

**Check pair with difference k**

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.

Sample Test Cases

 Test Case 1

 Input

 7

23

45

23

56

45

23

40

 Output

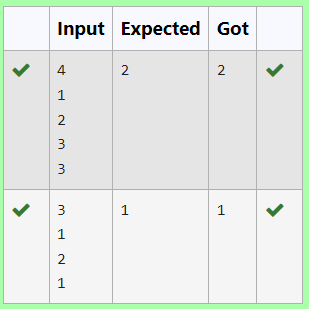
 23 occurs 3 times

45 occurs 2 times

56 occurs 1 times

40 occurs 1 times

**PROGRAM:**



**Ex. No. : 5.3 Date:**

**Register No.: 231801024 Name: Boopesh.R**

**Distinct Elements in an Array**

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

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:**

a=int(input())

c=set()

for i in range(a):

b=int(input())

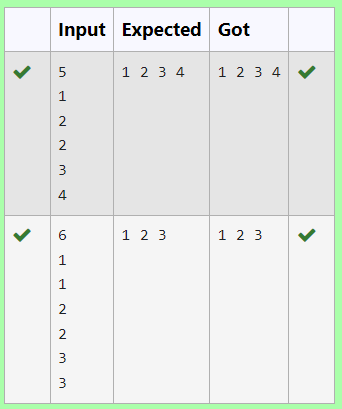
c.add(b)

d=list(c)

for j in range(len(d)):

print(d[j],end=" ")

**OUTPUT:**

****

**Ex. No. : 5.4 Date:**

**Register No.: 231801024 Name: Boopesh.R**

**Element Insertion**

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 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 = 3rd 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.

**For example:**

| **Input** | **Result** |
| --- | --- |
| 10  3 | 5 |
| 10  5 | 0 |
| 1  1 | 1 |

**PROGRAM:**

a=[]

for i in range(0,10):

e=int(input())

a.append(e)

c=int(input())

print("ITEM to be inserted:",c,sep="")

print("After insertion array is:")

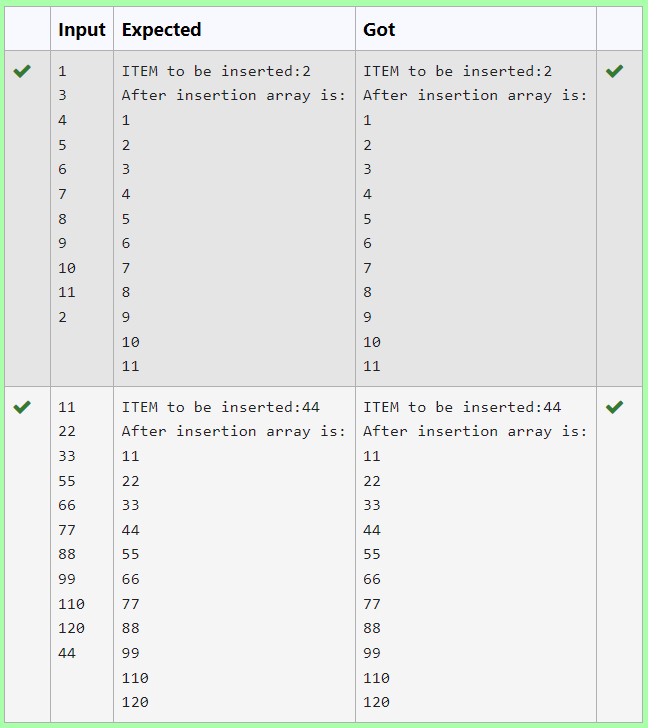
a.append(c)

a.sort()

for k in range(len(a)):

print(a[k])

**OUTPUT:**

****

**Ex. No. : 5.5 Date:**

**Register No.: 231801024 Name: Boopesh.R**

**Find the Factor**

Determine the factors of a number (i.e., all positive integer values that evenly divide into a number) and then return the pth element of the [list](http://118.185.187.137/moodle/mod/resource/view.php?id=732" \o "List), sorted ascending. If there is no pth element, return 0.

**Constraints**

1 ≤ n ≤ 1015

1 ≤ p ≤ 109

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 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:**

a=int(input())

b=int(input())

c=[]

for i in range(1,a+1):

if a%i==0:

c.append(i)

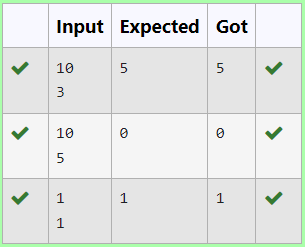
if b<=len(c):

print(c[b-1])

else:

print("0")

**OUTPUT:**

****

**Ex. No. : 5.6 Date:**

**Register No.: 231801024 Name: Boopesh.R**

**Merge List**

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 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:**

m=int(input())

n=int(input())

l1=[]

l2=[]

c=1

for I in range(0,m\*n\*2,2):

a=int(input())

b=int(input())

if c%2!=0:

l1.append(a)

l1.append(b)

else:

l2.append(a)

l2.append(b)

c+=1

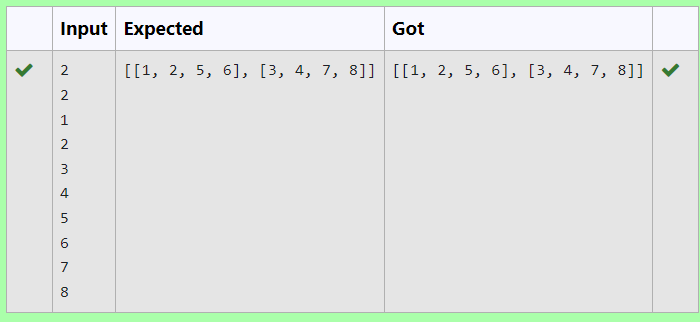
l3=[]

l3.append(l1)

l3.append(l2)

print(l3)

**OUTPUT:**

****

**Ex. No. : 5.7 Date:**

**Register No.: 231801024 Name: Boopesh.R**

**Merge Two Sorted Arrays Without Duplication**

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

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=[]

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:

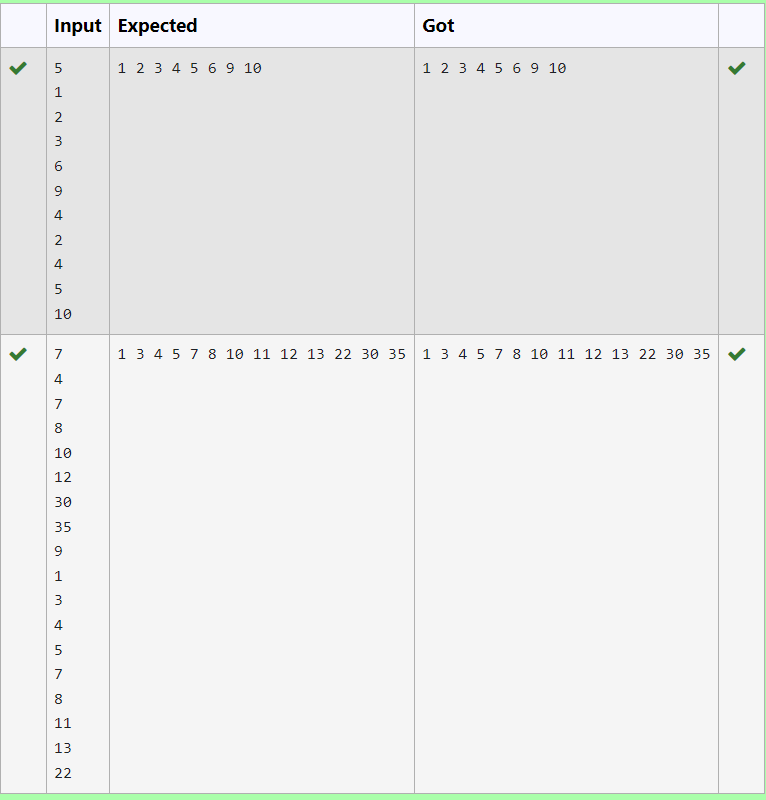
a.append(b[k])

a.sort()

for l in range(len(a)):

print(a[l],end=" ")

**OUTPUT:**

****

**Ex. No. : 5.8 Date:**

**Register No.: 231801024 Name: Boopesh.R**

**Print Element Location**

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.

Sample Test Case

Input

7

1

2

3

0

4

5

6

Output

True

**PROGRAM:**

n=int(input())

a=[]

for i in range(n):

b=int(input())

a.append(b)

c=int(input())

x=0

for i in range(len(a)):

if c==a[i]:

e=a.index(c)

print(c,"is present at location {}.".format(i+1))

x+=1

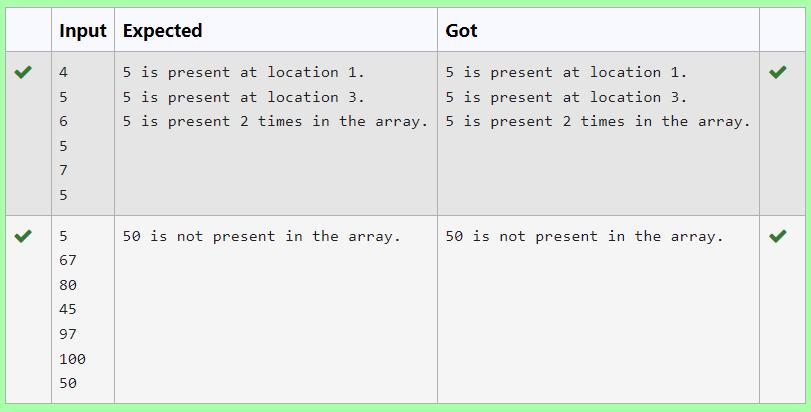
if(x>0):

print(c,"is present",x,"times in the array.")

else:

print(c,"is not present in the array.")

**OUTPUT:**

****

**Ex. No. : 5.9 Date:**

**Register No.: 231801024 Name: Boopesh.R**

**Strictly increasing**

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"

**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

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)

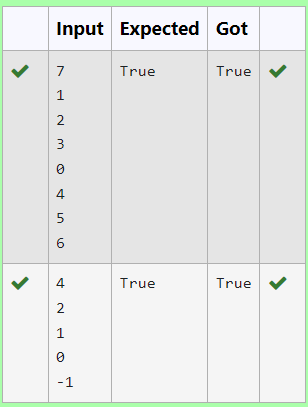
if is\_strictly\_increasing(list1) or is\_strictly\_decreasing(list1):

print("True")

else:

print("False")

**OUTPUT:**

****