05 - List in Python

Sample Case 0
Sample Input 0
4
1
2
3
3
Sample Output 0
2

Explanation 0
· The sum of the first two elements, 1+2=3. The value of the last element is 3.
· Using zero based indexing, arr[2]=3 is the pivot between the two subarrays.
· The index of the pivot is 2.

Sample Case 1
Sample Input 1
3
1
2
1
Sample Output 1
1
Explanation 1
· The first and last elements are equal to 1.
· Using zero based indexing, arr[1]=2 is the pivot between the two subarrays.
· The index of the pivot is 1.

For example:

Input	Result
4 1 2 3 3	2
3 1 2 1	1

Ex. No. : 5.1 Date:

Register No.: Name:

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

- $3 \leq n \leq 10^5$
- $1 \leq arr[i] \leq 2 \times 10^4$, where $0 \leq i < n$
- 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 $0 \leq i < n$.

```
a=int(input())
l=[]
for i in range(a):
   l.append(int(input()))
c=sum(l)//2
q=0
for j in l:
   q+=j
   if q >=c:
      q=j
      break;
print(l.index(q))
```

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

Ex. No. : 5.2 Date:

Register No.: Name:

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.

```
a=int(input())
for _ in range(a):
    l=[]
    s=0
    n = int(input())
    for _ in range(n):
        l.append(int(input()))
    k=int(input())
    for i in range(n):
        for j in range(i+1,n):
            if l[j]-l[i]==k and i!=j:
                s=1
                break
        if(s):
            break
    print(s)
```

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

Ex. No. : 5.3 Date:

Register No.: Name:

Count Elements

Complete the program to count frequency of each element of an array. Frequency of a particular element will be printed once.

```
a=int(input())
dic={}
for i in range(a):
    k=int(input())
    if k not in dic:
      dic[k]=1
    else:
      dic[k]+=1
for i in dic:
    print(f"{i} occurs {dic[i]} times")
```

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

For example:

Input	Result
5 1 2 2 3 4	1 2 3 4
6 1 1 2 2 3 3	1 2 3

Ex. No. : 5.4 Date:

Register No.: Name:

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

```
a=set()
b=int(input())
for i in range(b):
    a.add(int(input()))
b=sorted(list(a))
for i in b:
    print(i,end=' ')
```

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

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

Register No.: **Name:**

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.

```
l=[]
for i in range(10):
    l.append(int(input()))
c=int(input())
l.append(c)
l=sorted(l)
print(f"ITEM to be inserted:{c}\nAfter insertion array is:")
for i in l:
    print(i)
```

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

Ex. No. : 5.6 Date:

Register No.: Name:

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, sorted ascending. If there is no pth element, return 0.

Constraints

$1 \leq n \leq 1015$

$1 \leq p \leq 109$

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.

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

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]]

Ex. No. : 5.7 Date:

Register No.: Name:

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

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

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

Ex. No. : 5.8 Date:

Register No.: Name:

Merge Two Sorted Arrays Without Duplication

Output is a merged array without duplicates.

```
N1 = int(input())
array1 = [int(input()) for _ in range(N1)]
N2 = int(input())
array2 = [int(input()) for _ in range(N2)]
merged_array = array1 + array2

merged_array = list(set(merged_array))
merged_array.sort()
print(*merged_array)
```

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.

Ex. No. : 5.9 Date:

Register No.: Name:

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.

```
 num_elements = int(input())
lst = []
for _ in range(num_elements):
    lst.append(input())
element = input()
count = 0
for i in range(num_elements):
    if lst[i] == element:
        print(f"{element} is present at location {i + 1}.")
        count += 1
print(f"{element} is present {count} times in the array.")
```

Sample Test Case

Input

7

1

2

3

0

4

5

6

Output

True

Ex. No. : 5.10 Date:

Register No.: Name:

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"

```
def check_increasing_or_decreasing(lst):
    # Function to check if a list is strictly increasing or strictly decreasing
    increasing = True
    decreasing = True
    for i in range(1, len(lst)):
        if lst[i] > lst[i - 1]:
            decreasing = False
        elif lst[i] < lst[i - 1]:
            increasing = False
    return increasing or decreasing

def check_strictly_increasing_with_removal(lst):
    # Function to check if removing only one element makes the list strictly increasing or decreasing
    for i in range(len(lst)):
        temp_lst = lst[:i] + lst[i+1:]
        if check_increasing_or_decreasing(temp_lst):
            return True
```

```
    return False

# Input
n = int(input())
lst = []
for _ in range(n):
    lst.append(int(input()))

# Check if the list is strictly increasing or decreasing
if check_increasing_or_decreasing(lst) or check_strictly_increasing_with_removal(lst):
    print("True")
else:
    print("False")
```