

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, $\text{arr}[2]=3$ 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, $\text{arr}[1]=2$ 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

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

```
def find_pivot_index(arr):  
    total_sum = sum(arr)  
    left_sum = 0  
  
    for i in range(len(arr)):  
        total_sum -= arr[i]  
  
        if left_sum == total_sum:  
            return i  
  
        left_sum += arr[i]  
  
    # If no pivot is found, return -1  
    return -1  
  
# Reading input  
n = int(input())  
arr = [int(input()) for _ in range(n)]  
  
# Finding the pivot index  
pivot_index = find_pivot_index(arr)  
print(pivot_index)
```

Input

1

3

1

3

5

4

Output:

1

Input

1

3

1

3

5

99

Output

0

For example:

Input	Result
1	1
3	
1	
3	
5	
4	
1	0
3	
1	
3	
5	
99	

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 $A[i] - A[j] = k$, $i \neq j$.

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.

```
t=int(input())
while(t>0):
    n=int(input())
    a=[]
    count=0
    for i in range(0,n):
        s=int(input())
        a.append(s)
        k=int(input())
        for i in range(0,n):
            for j in range(0,n):
                if(i!=j):
                    if a[i]-a[j]==k:
                        count+=1
                        break

    if count==0:
        print("0")
    else:
        print("1")
    t-=1
```

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.

```
from collections import Counter
```

```
n = int(input())
```

```
a = []
```

```
for i in range(n):
```

```
    s = int(input())
```

```
    a.append(s)
```

```
c= Counter(a)
```

```
for e, f in c.items():
```

```
    print(f"{e} occurs {f} 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

```
t=int(input())
while t:
    t-=1
    n=int(input())
    nums=[int(input()) for i in range(n)]
    k=int(input())
    flag=False
    for i in nums:
        comp=abs(i-k)
        if comp in nums:
            flag=True
            break
    if flag:
        print(1)
    else:
        print(0)
```

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.

```
a=[]
b=11
for i in range(b):
    elem=int(input())
    a.append(elem)
c=sorted(a)
print("ITEM to be inserted:{}".format(a[10]))
print("After insertion array is:")
for num in c:
    print(num)
```

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 $p = 3^{\text{rd}}$ 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 $p = 1^{\text{st}}$ 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 p^{th} element of the [list](#), sorted ascending. If there is no p^{th} element, return 0.

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.

```
n = int(input().strip())
```

```
p = int(input().strip())
```

```
factors = []
```

```
for i in range(1, int(n**0.5) + 1):
```

```
    if n % i == 0:
```

```
        factors.append(i)
```

```
        if n // i != i:
```

```
            factors.append(n // i)
```

```
factors.sort()
```

```
if p <= len(factors):
```

```
    print (factors [p - 1])
```

```
else:
```

```
    print(0)
```


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

Zippped List : List which combined both list1 and list2

```
m = int(input())
```

```
n = int(input())
```

```
list1 = []
```

```
list2 = []
```

```
for _ in range(m):
```

```
    row = []
```

```
    for _ in range(n):
```

```
        row.append(int(input()))
```

```
    list1.append(row)
```

```
for _ in range(m):
```

```
    row = []
```

```
    for _ in range(n):
```

```
        row.append(int(input()))
```

```
    list2.append(row)
```



```
z= [row1 + row2 for row1, row2 in zip(list1, list2)]
```

```
print(z)
```

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.

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

```
b=int(input())
a=[]
c=[]
d=[]
for i in range(b):
    elem=int(input())
    a.append(elem)
f=int(input())
for j in range(f):
    elem=int(input())
    c.append(elem)
e=a+c
for num in e:
    if num not in d:
        d.append(num)
g=sorted(d)
for num in g:
    print(num,end=' ')
```

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.

```
a=[]
b=int(input())
for i in range(b):
    elem=int(input())
    a.append(elem)
c=int(input())
count=0
d=len(a)-1
for i in range(b):
    if(a[i]==c):
        count+=1
        print(c,"is present at location {}".format(i+1))

if(count==0):
    print(c,"is not present in the array.")
else:
    print(c,"is present {} times in the array.".format(count))
```

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"

```
n=int(input())
a=[int(input()) for x in range(n)]
def n1(l):
    i=all(l[i]<l[i+1] for i in range(len(l)-1))
    d=all(l[i]>l[i+1] for i in range(len(l)-1))
    return i or d
if n1(a):
    print("True")
else:
    for i in range(len(a)):
        t=a[:i]+a[i+1:]
        if n1(t):
            print("True")
            break
    else:
        print("False")
```