

Ex. No. : 6.1

Date: 04.05.24

Register No.:230701214

Name: Nithish Raj M

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 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:

```
x=[]  
for i in range(0,11):  
    b=int(input())  
    x.append(b)  
#a.sort()  
print("ITEM to be inserted:",x[-1],sep="")  
x.sort()  
print("After insertion array is:")  
for i in x:  
    print(i)
```



Ex. No. : 6.2

Date: 04.05.24

Register No.:230701214

Name: Nithish Raj M

Anagram

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 i th 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.

Note:

1. A, B have equal lengths in range [1, 100].
2. $A[i]$, $B[i]$ are integers in range $[0, 10^5]$.



Program:

```
def index_mapping(A, B):  
    index_map = {num: i for i, num in enumerate(B)}  
    return ''.join(str(index_map[num]) for num in A)  
  
n=int(input())  
A = list(map(int, input().split()))  
B = list(map(int, input().split()))  
print(index_mapping(A, B))
```



Ex. No. : 6.3

Date: 04.05.24

Register No.: 230701214

Name: Nithish Raj M

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

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:

```
n1=int(input())
l1=[]
for i in range(0,n1):
    a=int(input())
    l1.append(a)
n2=int(input())
l2=[]
for i in range(0,n2):
    a=int(input())
    l2.append(a)
l3=[]
l3.extend(l1)
l3.extend(l2)
a=list(set(l3))
a.sort()
for i in a:
    print(i,end=' ') n1=int(input())
l1=[]
for i in range(0,n1):
    a=int(input())
    l1.append(a)
n2=int(input())
l2=[]
for i in range(0,n2):
    a=int(input())
    l2.append(a)
l3=[]
l3.extend(l1)
l3.extend(l2)
a=list(set(l3))
a.sort()
for i in a:
    print(i,end=' ')
```



Ex. No. : 6.4

Date: 04.05.24

Register No.: 230701214

Name: Nithish Raj M

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

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 3 4
1	
2	
2	
3	
4	
6	1 2 3
1	
1	
2	
2	
3	
3	

Program:

```
n = int(input())
arr = []
for _ in range(n):
    arr.append(int(input()))
distinct_elements = set(arr)
print(*distinct_elements)
```



Ex. No. : 6.5

Date: 04.05.24

Register No.: 230701214

Name: Nithish Raj M

The Pivot

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 $0 \leq i < n$.

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



Program:

```
a = int(input())
b= []
for i in range(a):
    element = int(input())
    b.append(element)
total= sum(b)
left= 0
right = total- b[0]
if left== right:
    print(0)
    exit()
for i in range(1, a):
    left+= b[i - 1]
    right-= b[i]
    if left== right:
        print(i)
        break
```



Ex. No. : 6.6

Date: 04.05.24

Register No.: 230701214

Name: Nithish Raj M

Intersection of array

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

For example:

Input	Result
1 3 10 17 57 6 2 7 10 15 57 246	10 57
1 7 1 2 3 3 4 5 6 2 1 6	1 6



Program:

```
t=int(input())
l1=list()
while(t!=0):
    n1=int(input())
    l1=[]
    l2=[]
    for i in range(0,n1):
        a=int(input())
        l1.append(a)
    n2=int(input())
    for i in range(0,n2):
        a=int(input())
        l2.append(a)
    t=t-1
    c=set(l1)
    d=set(l2)
    e=list(c.intersection(d))
    e.sort()
    for i in e:
        print(i,end=' ')
    print("\n")
```



Ex. No. : 6.7

Date: 04.05.24

Register No.:230701214

Name: Nithish Raj M

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.

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:

```
n = int(input())
arr = [int(input()) for _ in range(n)]
element_to_search = int(input())
locations = []
occurrences = 0
for i in range(len(arr)):
    if arr[i] == element_to_search:
        locations.append(i + 1)
        occurrences += 1
if occurrences == 0:
    print(f"{element_to_search} is not present in the array.")
else:
    for loc in locations:
        print(f"{element_to_search} is present at location {loc}.")
    print(f"{element_to_search} is present {occurrences} times in the array.")
```



Ex. No. : 6.8

Date: 04.05.24

Register No.: 230701214

Name: Nithish Raj M

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"

Sample Test Case

Input

7

1

2

3

0

4

5

6

Output

True



Program:

```
def check_increasing_or_decreasing(lst):  
    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):  
    for i in range(len(lst)):  
        temp_lst = lst[:i] + lst[i+1:]  
        if check_increasing_or_decreasing(temp_lst):  
            return True  
    return False  
  
n = int(input())  
lst = []  
for _ in range(n):  
    lst.append(int(input()))
```



if check_increasing_or_decreasing(lst) or check_strictly_increasing_with_removal(lst):

 print("True")

else:

 print("False")

Ex. No. : 6.9

Date: 04.05.24

Register No.: 230701214

Name: Nithish Raj M

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

Zippered 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:

```
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=c+1
l3=[]
l3.append(l1)
l3.append(l2)
print(l3)
```



Ex. No. : 6.10

Date: 04.05.24

Register No.: 230701214

Name Nithish Raj M

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.

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	1



Input	Result
4	
1 3 1 3 5 99	0

Program:

```

t=int(input())

for i in range(0,t):

    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')
```

