WEEK 6

Experiments based on lists and its Operations

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]]

Answer:(penalty regime: 0 %)

```
m=int(input())
n=int(input())
list1=[]
for i in range(m):
    row=[]
    for j in range(n):
        element=int(input())
        row.append(element)
    list1.append(row)
list2=[]
for i in range(m):
    row=[]
    for j in range(n):
        element=int(input())
        row.append(element)
    list2.append(row)
zipped_list=[]
for i in range (m):
```

```
    zipped_row=list1[i]+list2[i]
    zipped_list.append(zipped_row)
print(zipped_list)
```

	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!

Correct

Marks for this submission: 1.00/1.00.

Question **2**
Correct
Mark 1.00 out of 1.00

Flag question

Question text

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.

Answer:(penalty regime: 0 %)

```
# Input array length
n = int(input())
# Input array elements
array = [int(input()) for _ in range(n)]
# Element to search
search_element = int(input())
# Initialize variables to store locations and count
locations = []
count = 0
# Search for the element and store its locations
for i in range(n):
    if array[i] == search_element:
        locations.append(i + 1)
        count += 1
# Print locations and count
if count > 0:
    for loc in locations:
        print(f"{search_element} is present at location {loc}.")
    print(f"{search_element} is present {count} times in the array.")
else:
    print(f"{search_element} is not present in the array.")
```

	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!

Correct

Marks for this submission: 1.00/1.00.

Question **3**
Correct
Mark 1.00 out of 1.00

Flag question

Question text

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

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

Answer:(penalty regime: 0 %)

```
# Input array length
```

```python
n = int(input())
# Input array elements
array = [int(input()) for _ in range(n)]
# Create a dictionary to store frequency of each element
frequency = {}
for num in array:
    if num in frequency:
        frequency[num] += 1
    else:
        frequency[num] = 1
# Print frequency of each element
for num, freq in frequency.items():
    print(f"{num} occurs {freq} times")
```

	Input	Expected	Got	
	7 23 45 23 56 45 23 40	23 occurs 3 times 45 occurs 2 times 56 occurs 1 times 40 occurs 1 times	23 occurs 3 times 45 occurs 2 times 56 occurs 1 times 40 occurs 1 times	

Passed all tests!

Correct

Marks for this submission: 1.00/1.00.

Question **4**
Correct
Mark 1.00 out of 1.00

Flag question

Question text

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

Answer:(penalty regime: 0 %)

```
# Input array length
n = int(input())
# Input array elements
array = [int(input()) for _ in range(n)]
# Create a set to store unique elements
distinct_elements = set(array)
# Print distinct elements separated by space
print(*distinct_elements)
```

	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!

Correct

Marks for this submission: 1.00/1.00.

Question **5**
Correct
Mark 1.00 out of 1.00

Flag question

Question text

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

Answer:(penalty regime: 0 %)

```
def merge_arrays(arr1, arr2):
    # If both arrays are empty, return an empty array
    if not arr1 and not arr2:
        return []

    # Create a set to store unique elements
    merged_set = set(arr1 + arr2)
    # Convert the set back to a sorted list to maintain order
    merged_array = sorted(list(merged_set))
    return merged_array
# Sample input
n1 = int(input())
array1 = [int(input()) for _ in range(n1)]
n2 = int(input())
array2 = [int(input()) for _ in range(n2)]
# Merge the arrays
merged = merge_arrays(array1, array2)
# Display the merged array
print(*merged)
```

	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	
	7 4 7 8 10 12 30 35 9 1	1 3 4 5 7 8 10 11 12 13 22 30 35	1 3 4 5 7 8 10 11 12 13 22 30 35	

	Input	Expected	Got	
	3 4 5 7 8 11 13 22			

Passed all tests!

Correct

Marks for this submission: 1.00/1.00.

Question **6**
Correct
Mark 1.00 out of 1.00

Flag question

Question text

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

For example:

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

Answer:(penalty regime: 0 %)

```
T=int(input())
for _ in range(T):
    N = int(input())
    A = [int(input()) for _ in range(N)]
    k = int(input())
    pair_exists = False
    for i in range(N):
        for j in range(i + 1 , N):
            if A[i] - A[j] == k or A[j] - A[i] == k:
                pair_exists = True
                break
        if pair_exists:
           break
    if pair_exists:
        print(1)
    else:
        print(0)
```

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

Passed all tests!

Correct

Marks for this submission: 1.00/1.00.

Question **7**
Correct
Mark 1.00 out of 1.00

Flag question

Question text

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

Answer:(penalty regime: 0 %)

```
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")
```

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

Passed all tests!

Correct

Marks for this submission: 1.00/1.00.

Question **8**
Correct
Mark 1.00 out of 1.00

Flag question

Question text

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 $0 \leq i < n$.

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

Answer:(penalty regime: 0 %)

```
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]

    return -1  # If no pivot is found
# Input processing
n = int(input())
arr = [int(input()) for _ in range(n)]
print(find_pivot_index(arr))
```

	Input	Expected	Got	
	4 1 2 3 3	2	2	
	3 1 2 1	1	1	

Passed all tests!

Correct

Marks for this submission: 1.00/1.00.

Question **9**
Correct
Mark 1.00 out of 1.00

Flag question

Question text

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.

For example:

Input	Result
10 3	5
10 5	0
1 1	1

Answer:(penalty regime: 0 %)

```
import math
def factor(n, p):
    factors = []
    sqrt_n = int(math.sqrt(n))

    for i in range(1, sqrt_n + 1):
        if n % i == 0:
            factors.append(i)
            if i != n // i:
                factors.append(n // i)

    factors.sort()

    if p > len(factors):
        return 0
    else:
        return factors[p - 1]
# Input processing
n = int(input())
p = int(input())
print(factor(n, p))
```

	Input	Expected	Got	
	10 3	5	5	

	Input	Expected	Got	
	10 5	0	0	
	1 1	1	1	

Passed all tests!

Correct

Marks for this submission: 1.00/1.00.

Question **10**
Correct
Mark 1.00 out of 1.00

Flag question

Question text

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

Answer:(penalty regime: 0 %)

```
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))
```

	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!

Correct

Marks for this submission: 1.00/1.00.