Started on	Friday, 24 May 2024, 8:28 AM
State	Finished
Completed on	Friday, 24 May 2024, 9:12 AM
Time taken	43 mins 27 secs
Marks	10.00/10.00
Grade	**100.00** out of 100.00

Question **1**

Correct

Mark 1.00 out of 1.00

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 %)

```python
# Creating an empty dictionary to store frequencies
frequency = {}

# Taking the number of elements as input
n = int(input())

# Taking input 'n' number of times and counting frequencies
for i in range(n):
    num = int(input())
    if num in frequency:
        frequency[num] += 1
    else:
        frequency[num] = 1

# Printing frequencies
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 **2**

Correct

Mark 1.00 out of 1.00

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 %)

```
n1=[]

t1= int(input())
for i in range(t1):
    elements = int(input())
    n1.append(elements)
    n1.append(elements)


merge_set = set (n1 )
merge_list= sorted(list(merge_set))




print(*merge_list)
```

	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 **3**

Correct

Mark 1.00 out of 1.00

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(""))
    l = []
    #print("Enter the elements of the list:")
    for _ in range(n):
```

```python
    l.append(int(input()))
k = int(input(""))

if n < 2:
    print("0")
else:
    l.sort()
    i = 0
    j = 1
    found = False
    while j < n and i < n:
        diff = l[j] - l[i]
        if i != j and diff == k:
            found = True
            break
        elif diff < k:
            j += 1
        else:
            i += 1
            if i == j:
                j += 1
if found:
    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 **4**

Correct

Mark 1.00 out of 1.00

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 %)

```
n1=[]
n2=[]

t1= int(input())
for i in range(t1):
    elements = int(input())
    n1.append(elements)

t2= int(input())
for j in range(t2):
    ele= int(input())
    n2.append(ele)

merge_set = set (n1 + n2)
merge_list= sorted(list(merge_set))




print(*merge_list)
```

	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 3 4 5 7 8 11 13 22	`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`	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Question **5**

Correct

Mark 1.00 out of 1.00

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 %)

```python
def find_pivot_index(arr):
    total_sum = sum(arr)
    left_sum = 0

    for i, num in enumerate(arr):
        total_sum -= num
        if left_sum == total_sum:
            return i
        left_sum += num

    # If no pivot is found, return -1 (this should not happen based on the problem description)
    return -1

# Reading input
def main():
    n = int(input().strip())
    arr = [int(input().strip()) for _ in range(n)]

    pivot_index = find_pivot_index(arr)
    print(pivot_index)

if __name__ == "__main__":
    main()
```

	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 **6**

Correct

Mark 1.00 out of 1.00

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

Answer: (penalty regime: 0 %)

```python
n = []

for i in range(0,10):
    ele = int(input())
    n.append(ele)

insert = int(input())
n.append(insert)
n.sort()
print(f"ITEM to be inserted:{insert}")
print("After insertion array is:")
print(*n, sep = "\n")
```

	Input	Expected	Got	
✓	1 3 4 5 6 7 8 9 10 11 2	ITEM to be inserted:2 After insertion array is: 1 2 3 4 5 6 7 8 9 10 11	ITEM to be inserted:2 After insertion array is: 1 2 3 4 5 6 7 8 9 10 11	✓
✓	11 22 33 55 66 77 88 99 110 120 44	ITEM to be inserted:44 After insertion array is: 11 22 33 44 55 66 77 88 99 110 120	ITEM to be inserted:44 After insertion array is: 11 22 33 44 55 66 77 88 99 110 120	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Question **7**

Correct

Mark 1.00 out of 1.00

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 %)

```python
def is_strictly_increasing(lst):
    # Check if the list is strictly increasing or strictly decreasing
    increasing = all(lst[i] < lst[i + 1] for i in range(len(lst) - 1))
    decreasing = all(lst[i] > lst[i + 1] for i in range(len(lst) - 1))

    return increasing or decreasing

def is_strictly_increasing_with_one_removed(lst):
    # Check if removing any element results in a strictly increasing list
    for i in range(len(lst)):
        temp_list = lst[:i] + lst[i+1:]
        if is_strictly_increasing(temp_list):
            return True

    return False

# Test Case
if __name__ == "__main__":
    n = int(input())  # Number of elements
    lst = [int(input()) for _ in range(n)]  # List of values

    # Check if the list is strictly increasing, strictly decreasing, or if removing one element result
    if is_strictly_increasing(lst) or is_strictly_increasing_with_one_removed(lst):
        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

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 %)

```python
def zip_lists(list1, list2):
    zipped_list = list(zip(list1, list2))
    return [sum(sublist, []) for sublist in zipped_list]

# Input
m = int(input())
n = int(input())

list1 = []
for _ in range(m):
    sublist = []
    for _ in range(n):
        sublist.append(int(input()))
    list1.append(sublist)

list2 = []
for _ in range(m):
    sublist = []
    for _ in range(n):
        sublist.append(int(input()))
    list2.append(sublist)

# Output
zipped_list = zip_lists(list1, list2)
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 **9**

Correct

Mark 1.00 out of 1.00

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 %)

```
times= int(input())
n = []
count =0
for i in range(times):
    ele = int(input())
    n.append(ele)

search = int(input())

for i in range(len(n)):
    if search == n[i]:
```

```
    if search == n[i]:
        print(f"{search} is present at location {i+1}.")
        count +=1

if count !=0:
    print(f"{search} is present {count} times in the array.")

else:
    print(f"{search} 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 **10**

Correct

Mark 1.00 out of 1.00

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

$1 \le n \le 10^{15}$

$1 \le p \le 10^{9}$

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 = 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`

Input	Result
1 1	1

Answer: (penalty regime: 0 %)

```
n= int(input())
factor= []

for i in range(1,n+1):
    if n%i ==0:
        factor.append(i)


p = int(input())

if p > len(factor):
    print("0")

elif p <= len(factor):
    print(factor[p-1])




```

	Input	Expected	Got	
✓	10 3	5	5	✓
✓	10 5	0	0	✓
✓	1 1	1	1	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.