| | |
|---|---|
| **Started on** | Thursday, 23 May 2024, 2:29 PM |
| **State** | Finished |
| **Completed on** | Friday, 24 May 2024, 8:31 AM |
| **Time taken** | 18 hours 2 mins |
| **Marks** | 10.00/10.00 |
| **Grade** | **100.00** out of 100.00 |

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
1  def count_freq(arr):
2      freq_dict={}
3      for element in arr:
4          if element in freq_dict:
5              freq_dict[element]+=1
6          else:
7              freq_dict[element]=1
8      for key in (freq_dict.keys()):
9          print(f"{key} occurs {freq_dict[key]} times")
10 n=int(input())
11 arr=[]
12 for _ in range(n):
13     arr.append(int(input()))
14
15 count_freq(arr)
```

| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | 7<br>23<br>45<br>23<br>56<br>45<br>23<br>40 | 23 occurs 3 times<br>45 occurs 2 times<br>56 occurs 1 times<br>40 occurs 1 times | 23 occurs 3 times<br>45 occurs 2 times<br>56 occurs 1 times<br>40 occurs 1 times | ✓ |

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

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<br>1<br>2<br>3<br>3 | 2 |
| 3<br>1<br>2<br>1 | 1 |

**Answer:**  (penalty regime: 0 %)

```python
1   n = int(input())
2   arr = []
3   for _ in range(n):
4       arr.append(int(input()))
5
6
7       total_sum = sum(arr)
8       left_sum = 0
9       for i in range(len(arr)):
10          total_sum -= arr[i]
11
12          if (left_sum == total_sum and left_sum!=0):
13
14              print(i)
15          left_sum += arr[i]
16
17
18
```

| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | 4<br>1<br>2<br>3<br>3 | 2 | 2 | ✓ |
| ✓ | 3<br>1<br>2<br>1 | 1 | 1 | ✓ |

Passed all tests!  ✓

Correct

Marks for this submission: 1.00/1.00.

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
1   arr = []
2   #print("Enter the sorted array (size 10):")
3   for _ in range(10):
4       arr.append(int(input()))
5
6   item = int(input("ITEM to be inserted:"))
7   print(item)
8   index = 0
9   while index < len(arr) and arr[index] < item:
10      index += 1
11  arr.insert(index, item)
12
13  print("After insertion array is:")
14  for element in arr:
15      print(element)
```

| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | 1<br>3<br>4<br>5<br>6<br>7<br>8<br>9<br>10<br>11<br>2 | ITEM to be inserted:2<br>After insertion array is:<br>1<br>2<br>3<br>4<br>5<br>6<br>7<br>8<br>9<br>10<br>11 | ITEM to be inserted:2<br>After insertion array is:<br>1<br>2<br>3<br>4<br>5<br>6<br>7<br>8<br>9<br>10<br>11 | ✓ |

| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | 11<br>22<br>33<br>55<br>66<br>77<br>88<br>99<br>110<br>120<br>44 | ITEM to be inserted:44<br>After insertion array is:<br>11<br>22<br>33<br>44<br>55<br>66<br>77<br>88<br>99<br>110<br>120 | ITEM to be inserted:44<br>After insertion array is:<br>11<br>22<br>33<br>44<br>55<br>66<br>77<br>88<br>99<br>110<br>120 | ✓ |

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

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 = $3^{rd}$ 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<br>3 | 5 |
| 10<br>5 | 0 |
| 1<br>1 | 1 |

**Answer:**  (penalty regime: 0 %)

```python
def factors_sorted(n, p):
    factors = []
    for i in range(1, int(n ** 0.5) + 1):
        if n % i == 0:
            factors.append(i)
            if i != n // i:
                factors.append(n // i)
    factors.sort()
    if p > len(factors):
        return 0
    return factors[p - 1]
n = int(input())
p = int(input())
print(factors_sorted(n, p))
```

| | Input | Expected | Got | |
|---|-------|----------|-----|---|
| ✓ | 10<br>3 | 5 | 5 | ✓ |
| ✓ | 10<br>5 | 0 | 0 | ✓ |
| ✓ | 1<br>1 | 1 | 1 | ✓ |

Passed all tests! ✓

Marks for this submission: 1.00/1.00.

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<br>3<br>1<br>3<br>5<br>4 | 1 |
| 1<br>3<br>1<br>3<br>5<br>99 | 0 |

**Answer:** (penalty regime: 0 %)

```python
def find_pair_with_difference(arr, k):
    seen = set()
    for num in arr:
        if num - k in seen:
            return 1
        seen.add(num)
    return 0

T = int(input())

for _ in range(T):
    N = int(input())
    arr = []
    for _ in range(N):
        arr.append(int(input()))
    k = int(input())

    # Check if pair with difference k exists
    result = find_pair_with_difference(arr, k)
    print(result)
```

| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | 1<br>3<br>1<br>3<br>5<br>4 | 1 | 1 | ✓ |
| ✓ | 1<br>3<br>1<br>3<br>5<br>99 | 0 | 0 | ✓ |

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

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
1  def zip_lists(list1, list2):
2      zipped_list = list(zip(list1, list2))
3      return [sum(sublist, []) for sublist in zipped_list]
4
5  # Input
6  m = int(input())
7  n = int(input())
8
9  list1 = []
10 for _ in range(m):
11     sublist = []
12     for _ in range(n):
13         sublist.append(int(input()))
14     list1.append(sublist)
15
16 list2 = []
17 for _ in range(m):
18     sublist = []
19     for _ in range(n):
20         sublist.append(int(input()))
21     list2.append(sublist)
22
23 # Output
24 zipped_list = zip_lists(list1, list2)
25 print(zipped_list)
```

| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | 2<br>2<br>1<br>2<br>3<br>4<br>5<br>6<br>7<br>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.

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
 1  def is_strictly_increasing(lst):
 2      for i in range(len(lst) - 1):
 3          if lst[i] >= lst[i + 1]:
 4              if i == 0 or lst[i - 1] < lst[i + 1] or i == len(lst) - 2 or lst[i] < lst[i + 2]:
 5                  return True
 6              else:
 7                  return False
 8      return True
 9  n = int(input())
10  lst = [int(input()) for _ in range(n)]
11  if is_strictly_increasing(lst):
12      print("True")
13  else:
14      print("False")
```

| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | 7<br>1<br>2<br>3<br>0<br>4<br>5<br>6 | True | True | ✓ |
| ✓ | 4<br>2<br>1<br>0<br>-1 | True | True | ✓ |

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

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<br>1<br>2<br>2<br>3<br>4 | 1 2 3 4 |
| 6<br>1<br>1<br>2<br>2<br>3<br>3 | 1 2 3 |

**Answer:** (penalty regime: 0 %)

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

|  | Input | Expected | Got |  |
|---|---|---|---|---|
| ✓ | 5<br>1<br>2<br>2<br>3<br>4 | 1 2 3 4 | 1 2 3 4 | ✓ |
| ✓ | 6<br>1<br>1<br>2<br>2<br>3<br>3 | 1 2 3 | 1 2 3 | ✓ |

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

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 %)

```
1  def find_element_locations(lst, element):
2      locations = []
3      count = 0
```

```
 4 ▾        for i, num in enumerate(lst, start=1):
 5 ▾            if num == element:
 6                  locations.append(i)
 7                  count += 1
 8
 9          return locations, count
10     n = int(input())
11     lst = [int(input()) for _ in range(n)]
12     element = int(input())
13
14     locations, count = find_element_locations(lst, element)
15
16 ▾   if count == 0:
17          print(f"{element} is not present in the array.")
18 ▾   else:
19 ▾        for loc in locations:
20              print(f"{element} is present at location {loc}.")
21          print(f"{element} is present {count} times in the array.")
```

| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | 4<br>5<br>6<br>5<br>7<br>5 | 5 is present at location 1.<br>5 is present at location 3.<br>5 is present 2 times in the array. | 5 is present at location 1.<br>5 is present at location 3.<br>5 is present 2 times in the array. | ✓ |
| ✓ | 5<br>67<br>80<br>45<br>97<br>100<br>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.

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 %)

```python
def merge_and_remove_duplicates(arr1, arr2):
    merged_set = set(arr1 + arr2)
    merged_list = sorted(list(merged_set))
    print(*merged_list)

# Input
n1 = int(input())
arr1 = []
for _ in range(n1):
    arr1.append(int(input()))

n2 = int(input())
arr2 = []
for _ in range(n2):
    arr2.append(int(input()))

# Output
merge_and_remove_duplicates(arr1, arr2)
```

| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | 5<br>1<br>2<br>3<br>6<br>9<br>4<br>2<br>4<br>5<br>10 | 1 2 3 4 5 6 9 10 | 1 2 3 4 5 6 9 10 | ✓ |
| ✓ | 7<br>4<br>7<br>8<br>10<br>12<br>30<br>35<br>9<br>1<br>3<br>4<br>5<br>7<br>8<br>11<br>13<br>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.

Jump to...