10 – Search Techniques

Ex. No.: 10.1 Date:

Register No.: 2116231501105 Name: Nandhini Prakash

To find the frequency of numbers in a list and display in sorted order.

Constraints:

1<=n, arr[i]<=100

Input:

1 68 79 4 90 68 1 4 5

output:

1 2

4 2

5 1

68 2

79 1

90 1

For example:

Input	Result
4 3 5 3 4 5	3 2 4 2 5 2

```python
n = list(map(int, input().split()))
d = {}
for i in n:
    if i not in d:
        d[i] = 1
    else:
        d[i] = d[i] + 1
l = [i for i in d.keys()]
l.sort()
for i in l:
    if i in d:
        print(i, d[i])

```

Output:

✔	5 4 5 4 6 5 7 3	3 1 4 2 5 3 6 1 7 1	3 1 4 2 5 3 6 1 7 1	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

Ex. No.: 10.2 Date:

Register No.: 2116231501105 Name: Nandhini Prakash

Given an list, find peak element in it. A peak element is an element that is greater than its neighbors.

An element a[i] is a peak element if

A[i-1] <= A[i] >=a[i+1] for middle elements. [0<i<n-1]

A[i-1] <= A[i] for last element [i=n-1]

A[i]>=A[i+1] for first element [i=0]

Input Format

The first line contains a single integer n , the length of A .
The second line contains n space-separated integers,A[i].

Output Format

Print peak numbers separated by space.

```
n = int(input())
a = list(map(int, input().split()))
if(a[0]>a[1]):
    print(a[0],end = " ")
for i in range(1, n-1):
    if(a[i-1]<a[i] and a[i]>a[i+1]):
        print(a[i],end = " ")
if(a[-1]>a[-2]):
    print(a[-1])
```

Output:

	Input	Expected	Got	
✔	7 15 7 10 8 9 4 6	15 10 9 6	15 10 9 6	✔
✔	4 12 3 6 8	12 8	12 8	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00

Ex. No.: 9.3 Date:

Register No.: 2116231501105 Name: Nandhini Prakash

Given an listof integers, sort the array in ascending order using the *Bubble Sort* algorithm above. Once sorted, print the following three lines:

1. List is sorted in numSwaps swaps., where numSwaps is the number of swaps that took place.
2. First Element: firstElement, the *first* element in the sorted list.
3. Last Element: lastElement, the *last* element in the sorted list.

For example, given a worst-case but small array to sort: a=[6,4,1]. It took 3 swaps to sort the array. Output would be

```
Array is sorted in 3 swaps.

First Element: 1

Last Element: 6
```

Program:

```
def bubble_sort(arr):
    n = len(arr)
    numSwaps = 0
    for i in range(n):
        for j in range(0, n-i-1):
            if arr[j] > arr[j+1]:
                arr[j], arr[j+1] = arr[j+1], arr[j]
                numSwaps += 1
    return numSwaps, arr
n = int(input())
a = list(map(int, input().split()))
numSwaps, sorted_array = bubble_sort(a)
print(f"List is sorted in {numSwaps} swaps.")
print(f"First Element: {sorted_array[0]}")
print(f"Last Element: {sorted_array[-1]}")

```

Output:

	Input	Expected	Got	
✔	3 3 2 1	List is sorted in 3 swaps. First Element: 1 Last Element: 3	List is sorted in 3 swaps. First Element: 1 Last Element: 3	✔
✔	5 1 9 2 8 4	List is sorted in 4 swaps. First Element: 1 Last Element: 9	List is sorted in 4 swaps. First Element: 1 Last Element: 9	✔

Passed all tests! ✔

Correct

Ex. No.: 10.4 Date:

Register No.: 2116231501105 Name: Nandhini Prakash

Bubble Sort is the simplest sorting algorithm that works by repeatedly swapping the adjacent elements if they are in wrong order. You read an list of numbers. You need to arrange the elements in ascending order and print the result. The sorting should be done using bubble sort.

Input Format: The first line reads the number of elements in the array. The second line reads the array elements one by one.

Output Format: The output should be a sorted list.

For example:

Input	Result
6 3 4 8 7 1 2	1 2 3 4 7 8
5 4 5 2 3 1	1 2 3 4 5

```
def bubble_sort(arr):
    n = len(arr)
    for i in range(n):
        for j in range(0, n-i-1):
            if arr[j] > arr[j+1]:
                arr[j], arr[j+1] = arr[j+1], arr[j]

num_elements = int(input())
array = list(map(int, input().split()))

bubble_sort(array)

for element in array:
    print(element, end=" ")


```

Output:

	Input	Expected	Got	
✔	6 3 4 8 7 1 2	1 2 3 4 7 8	1 2 3 4 7 8	✔
✔	6 9 18 1 3 4 6	1 3 4 6 9 18	1 3 4 6 9 18	✔
✔	5 4 5 2 3 1	1 2 3 4 5	1 2 3 4 5	✔

Passed all tests! ✔

Correct

Ex. No.: 10.5 Date:

Register No.: 2116231501105 Name: Nandhini Prakash

Write a Python program to sort a list of elements using the merge sort algorithm.

For example:

Input	Result
5 6 5 4 3 8	3 4 5 6 8

Program:

```python
def merge_sort(arr):
    if len(arr) > 1:
        # Finding the mid of the array
        mid = len(arr) // 2

        # Dividing the elements into 2 halves
        left_half = arr[:mid]
        right_half = arr[mid:]

        # Recursively sort each half
        merge_sort(left_half)
        merge_sort(right_half)

        i = j = k = 0

        # Copy data to temporary arrays L[] and R[]
        while i < len(left_half) and j < len(right_half):
            if left_half[i] < right_half[j]:
                arr[k] = left_half[i]
                i += 1
            else:
                arr[k] = right_half[j]
                j += 1
            k += 1

        # Checking if any element was left
        while i < len(left_half):
            arr[k] = left_half[i]
            i += 1
            k += 1

        while j < len(right_half):
            arr[k] = right_half[j]
            j += 1
            k += 1

# Input list
n = int(input())
a = list(map(int, input().split()))

# Calling merge_sort function
merge_sort(a)
```

Output:

	Input	Expected	Got
✔	5 6 5 4 3 8	3 4 5 6 8	3 4 5 6 8
✔	9 14 46 43 27 57 41 45 21 70	14 21 27 41 43 45 46 57 70	14 21 27 41 43 45 46 57
✔	4 86 43 23 49	23 43 49 86	23 43 49 86

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.