**WEEK-10 CODING**

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

\* List is sorted in numSwaps swaps., where numSwaps is the number of swaps that took place.

\* First Element: firstElement, the first element in the sorted list.

\* 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

Sample Input 0

3

1 2 3

Sample Output 0

List is sorted in 0 swaps.

First Element: 1

Last Element: 3

**PROGRAM :**

def count\_swaps\_and\_sort(A):

swaps = 0

n = len(A)

for i in range(n):

for j in range(n - 1):

if A[j] > A[j + 1]:

A[j], A[j + 1] = A[j + 1], A[j]

swaps += 1

return swaps, A

def find\_peaks(A):

peaks = []

p []

n = len(A)

# Check if the array has only one element

if n == 1:

peaks.append(A[0])

return peaks

# Check first element

if A[0] >= A[1]:

peaks.append(A[0])

# Check middle elements

for i in range(1, n-1):

if A[i] >= A[i-1] and A[i] >= A[i+1]:

peaks.append(A[i])

# Check last element

if A[n-1] >= A[n-2]:

peaks.append(A[n-1])

return peaks

# Reading input

n = int(input().strip())

A = list(map(int, input().strip().split()))

# Counting swaps and sorting the array

swaps, sorted\_A = count\_swaps\_and\_sort(A.copy())

# Printing sorting results

print(f"List is sorted in {swaps} swaps.")

print(f"First Element: {sorted\_A[0]}")

print(f"Last Element: {sorted\_A[-1]}")

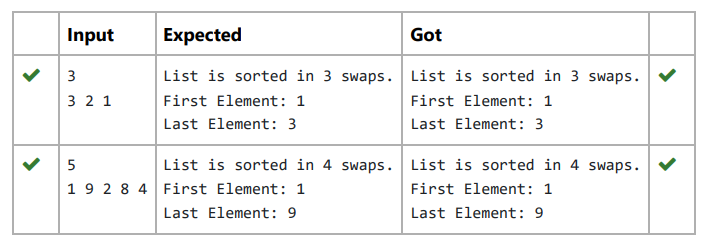
# Finding peaks

peaks = find\_peaks(A)

# (Optional) Printing peaks separately if needed

# print(" ".join(map(str, peaks)))

**OUTPUT**

****

2.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.

Sample Input

5

8 9 10 2 6

Sample Output

10 6

**PROGRAM:**

def find\_peaks(n, A):

peaks = []

# Check first element

if n > 0 and (n == 1 or A[0] >= A[1]):

peaks.append(A[0])

# Check middle elements

for i in range(1, n-1):

if A[i] >= A[i-1] and A[i] >= A[i+1]:

peaks.append(A[i])

# Check last element

if n > 1 and A[n-1] >= A[n-2]:

peaks.append(A[n-1])

return peaks

# Reading input

n = int(input().strip())

A = list(map(int, input().strip().split()))

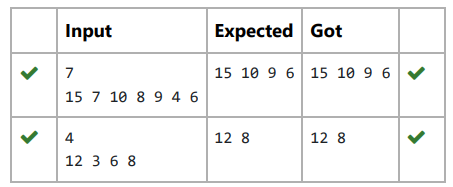
# Finding peaks

peaks = find\_peaks(n, A)

# Printing peaks

print(' '.join(map(str, peaks)))

**OUTPUT:**

****

3.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.

**PROGRAM:**

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]

def main():

n = int(input())

arr = list(map(int, input().split()))

bubble\_sort(arr)

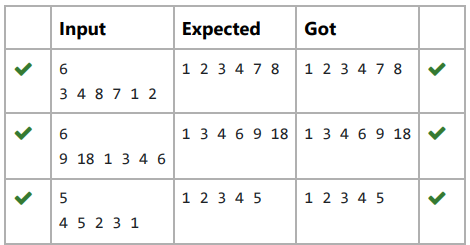
for num in arr:

print(num, end=" ")

if \_\_name\_\_ == "\_\_main\_\_":

main()

**OUTPUT:**

****

4.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

**PROGRAM :**

def count\_frequencies(arr):

frequency = {}

for num in arr:

if num in frequency:

frequency[num] += 1

else:

frequency[num] = 1

sorted\_frequency = sorted(frequency.items())

return sorted\_frequency

def main():

arr = list(map(int, input().split()))

sorted\_frequency = count\_frequencies(arr)

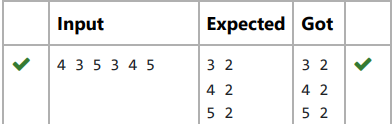
for num, freq in sorted\_frequency:

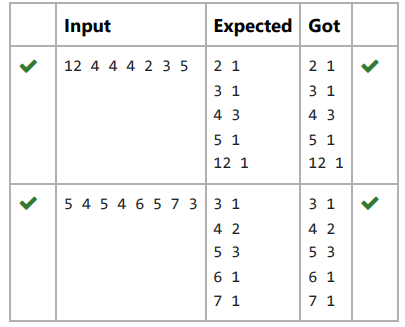
print(num, freq)

if \_\_name\_\_ == "\_\_main\_\_":

main()

**OUTPUT:**

****

****

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

**PROGRAM:**

def merge\_sort(arr):

if len(arr) > 1:

mid = len(arr) // 2

left\_half = arr[:mid]

right\_half = arr[mid:]

merge\_sort(left\_half)

merge\_sort(right\_half)

i = j = k = 0

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

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

def main():

n = int(input())

arr = list(map(int, input().split()))

merge\_sort(arr)

for num in arr:

print(num, end=" ")

if \_\_name\_\_ == "\_\_main\_\_":

main()

**OUTPUT:**

