10 - Searching & Sorting

Ex. No. : 10.1 **Date:** 25/5/24

Register No.: 231501053 **Name: Gokulakkannan P**

Merge Sort

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

```
a = int(input())
b = list(input().split(" "))
b.sort()
for i in b:
    print(i,end=" ")
```

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 5
✔	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.

Ex. No. : 10.2 **Date:** 25/5/24

Register No.: 231501053 **Name: Gokulakkannan P**

Bubble Sort

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

Input Format

The first line contains an integer,n , the size of the list a . The second line contains n, space-separated integers a[i].

Constraints

- 2<=n<=600
- 1<=a[i]<=$2x10^6$.

Output Format

You must print the following three lines of output:

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.

Sample Input 0

3

1 2 3

Sample Output 0

List is sorted in 0 swaps.

First Element: 1

Last Element: 3

For example:

Input	Result
3 3 2 1	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

PROGRAM

```
num = 0
a = int(input())
b = input().split(" ")
c = []

for i in range(len(b)):
    c.append(int(b[i]))

for j in range(len(c)):
    for i in range(len(c)-1):
        if c[i] > c[i+1]:
            c[i], c[i+1] = c[i+1], c[i]
            num += 1

print(f'List is sorted in {num} swaps.\nFirst Element: {c[0]}\nLast Element: {c[-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

Marks for this submission: 1.00/1.00.

Ex. No. : 10.3 **Date:** 25/5/24

Register No.: 231501053 **Name: Gokulakkannan P**

Peak Element

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

Input	Result
4 12 3 6 8	12 8

For example:

PROGRAM

```
n = int(input())
A = list(map(int, input().split()))
if n == 1:
    print(A[0])
else:
    if A[0] >= A[1]:
        print(A[0], end=" ")
    for i in range(1, n - 1):
        if A[i] >= A[i - 1] and A[i] >= A[i + 1]:
            print(A[i], end=" ")
    if A[n - 1] >= A[n - 2]:
        print(A[n - 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. : 10.4 **Date:** 25/5/24

Register No.: 231501053 **Name: Gokulakkannan P**

Binary Search

Write a Python program for binary search.

For example:

Input	Result
1 2 3 5 8 6	False
3 5 9 45 42 42	True

PROGRAM

```
def binary_search(arr, x):
    left, right = 0, len(arr) - 1
    while left <= right:
        mid = (left + right) // 2
        if arr[mid] == x:
            return True
        elif arr[mid] < x:
            left = mid + 1
        else:
            right = mid - 1
    return False


arr = list(map(int, input().split(',')))
x = int(input())
arr.sort()
result = binary_search(arr, x)
print(result)
```

Output:

	Input	Expected	Got	
✔	1,2,3,5,8 6	False	False	✔
✔	3,5,9,45,42 42	True	True	✔
✔	52,45,89,43,11 11	True	True	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

Ex. No. : 10.5 **Date:** 25/5/24

Register No.: 231501053 **Name: Gokulakkannan P**

Frequency of Elements

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

PROGRAM

```
numbers = input().split()
numbers = [int(num) for num in numbers]
frequency = {}
for num in numbers:
    if num in frequency:
        frequency[num] += 1

 else:
        frequency[num] = 1
sorted_frequency = sorted(frequency.items())
for key, value in sorted_frequency:
    print(key, value)
```

Output:

	Input	Expected	Got	
✔	4 3 5 3 4 5	3 2 4 2 5 2	3 2 4 2 5 2	✔
✔	12 4 4 4 2 3 5	2 1 3 1 4 3 5 1 12 1	2 1 3 1 4 3 5 1 12 1	✔
✔	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.