10 - Searching & Sorting

For example:

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

Ex. No. : 10.1 **Date:**

Register No.: **Name:**

Merge Sort

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

```
def merge(a,l,m,h):
    p=l
    q=m+1
    temp=[]
    while p<=m and q<=h:
        if(a[p]>a[q]):
            temp.append(a[q])
            q+=1
        else:
            temp.append(a[p])
            p+=1
    while p<=m:
        temp.append(a[p])
        p+=1
    while q<=h:
        temp.append(a[q])
        q+=1
    for i in range(len(temp)):
        a[l+i]=temp[i]
def mergesort(a,l,h):
    if(l==h):
        return
    m=(l+h)//2
    mergesort(a,l,m)
    mergesort(a,m+1,h)
    merge(a,l,m,h)
n=int(input())
a=list(eval(input().replace(' ',',')))
mergesort(a,0,len(a)-1)
for i in a:
    print(i,end=' ')
```

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

Ex. No. : 10.2 **Date:**

Register No.: **Name:**

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 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

For example:

Input	Result
4 12 3 6 8	12 8

Ex. No. : 10.3 **Date:**

Register No.: **Name:**

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]

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

For example:

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

Ex. No. : 10.4 **Date:**

Register No.: **Name:**

Binary Search

Write a Python program for binary search.

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

Ex. No. : 10.5 **Date:**

Register No.: **Name:**

Frequency of Elements

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

Constraints:

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

```
a=sorted([int(x) for x in input().split(' ')])
s={}
for i in a:
    if i in s:
        s[i]+=1
    else:
        s[i]=1
for i in s:
    print(i,s[i])
```