10 - Searching & Sorting

Ex. No. : 10.1 **Date: 18/05/24**

Register No.: 230701348 **Name: N SUBRAMANIAN**

Merge Sort

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

```
def merge_sort(arr):
    if len(arr)>1:
        mid=len(arr)//2
        L=arr[:mid]
        R=arr[mid:]
        merge_sort(L)
        merge_sort(R)
        i=j=k=0
        while i < len(L) and j <len(R):
            if L[i]<R[j]:
                arr[k]=L[i]
                i+=1
            else:
                arr[k]=R[j]
                j+=1
            k+=1

        while i <len(L):
            arr[k]=L[i]
            i+=1
            k+=1
        while j<len(R):
            arr[k]=R[j]
            j+=1
            k+=1
n=int(input())
arr=list(map(int,input().split()))
merge_sort(arr)
print(*arr)
```


For example:

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

Ex. No. : 10.2 **Date: 18/05/24**

Register No.: 230701348 **Name: N SUBRAMANIAN**

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
```

```
def bubbleSort(arr):
    n = len(arr)
    for i in range(n-1):
        for j in range(0, n-i-1):
            if arr[j] > arr[j + 1]:
                arr[j], arr[j + 1] = arr[j + 1], arr[j]
    return arr
n=int(input())
s=input().split()
s=[int(e) for e in s]
lst=bubbleSort(s)
for i in lst:
    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. : 10.3 **Date: 18/05/24**

Register No.: 230701348 **Name: N SUBRAMANIAN**

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


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. : 10.4 **Date: 18/05/24**

Register No.: 230701348 **Name: N SUBRAMANIAN**

Binary Search

Write a Python program for binary search.

```
lst=input().split(',')
for i in range(len(lst)):
    lst[i]=int(lst[i])
search=int(input())
def binary_search(l,k):
    flag=0
    low=0
    high=len(lst)
    while low<=high:
        mid=low+(high-low)//2
        if k==l[mid]:
            return "True"
        elif k>l[mid]:
            low=mid+1
        else:
            high=mid-1
    return "False"
print(binary_search(sorted(lst),search))
```


For example:

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

Ex. No. : 10.5 **Date: 18/05/24**

Register No.: 230701348 **Name: N SUBRAMANIAN**

Frequency of Elements

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

Constraints:

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

```
numbers=list(map(int,input().split()))
freq={}
for n in numbers:
    freq[n]=freq.get(n,0)+1
sorted_freq=sorted(freq.items())
for num,freq in sorted_freq:
    print(num,freq)
```


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