Ex. No. : 10.1

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

n=int(input())

l=input().split(" ")

for i in range (n):

  l[i]=int(l[i])

l.sort()

print(\*l)

Ex. No. : 10.2

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]<=2x106.

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

n=int(input())

l=input().split(" ")

count=0

for i in range (n-1,0,-1):

  flag=0

  for j in range(i):

  if(int(l[j])>int(l[j+1])):

  flag=1

  l[j],l[j+1]=l[j+1],l[j]

  count+=1

  if not flag:

  break

print("List is sorted in",count,"swaps.") print("First Element:",l[0])

print("Last Element",l[-1])

Ex. No. : 10.3

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

For example:

Input

Result

4

12 3 6 8

12 8

n=int(input())

l=input().split(" ")

l1=[]

for i in range(n):

  if(i>0 and i<n-1):

  if(int(l[i-1])<=int(l[i]) and int(l[i])>=int(l[i+1])): l1.append(int(l[i]))

  elif(i==0 and int(l[i])>=int(l[i+1])):

  l1.append(int(l[i]))

  elif(i==(n-1) and int(l[i])>=int(l[i-1])):

  l1.append(int(l[i]))

for i in l1:

  print(i,end=" ")

Ex. No. : 10.4

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

def binary\_search(arr, target):

    left, right = 0, len(arr) - 1

    while left <= right:

        mid = (left + right) // 2

        if arr[mid] == target:

            return True

        elif arr[mid] < target:

            left = mid + 1

        else:

            right = mid - 1

    return False

sorted\_list = list(map(int, input().split(',')))

target = int(input())

sorted\_list.sort()

result = binary\_search(sorted\_list, target)

print(result)

Ex. No. : 10.5

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

l=input().split(" ")

d={}

s=set()

for i in l:

  count=0

  if(i not in s):

  s.add(i)

  for j in l:

  if(i==j):

  count+=1

  d[int(i)]=count

l1=list(d.keys()) l1.sort()

for i in l1:

  print(i,d[i])