**Completed on** Sunday, 3 November 2024, 9:09 AM
**Time taken** 8 mins 36 secs

Question **1**
Complete
Marked out of 1.00
Flag question

What is a key disadvantage of Bubble Sort compared to more advanced algorithms like Merge Sort?

a. Bubble Sort is difficult to implement
b. Bubble Sort does not guarantee sorted order
c. Bubble Sort is less efficient for large lists
d. Bubble Sort cannot handle duplicate elements

Question **2**
Complete
Marked out of 1.00
Flag question

What does the Bubble Sort algorithm primarily focus on during each pass?

a. Sorting the entire list in one pass
b. Dividing the list into halves
c. Bubbling up the largest element to its correct position
d. Bubbling up the smallest element

Question **3**
Complete
Marked out of 1.00
Flag question

What is one of the key advantages of using the built-in sorted() function in Python?

a. It only works with integer arrays
b. It sorts data out of the box efficiently
c. It requires external libraries
d. It is less efficient than custom sorting algorithms

Question **4**
Complete
Marked out of 1.00
Flag question

What is Bubble Sort known for?

a. Sorting data in a non-sequential manner
b. Being the most efficient sorting algorithm
c. Bubbling up the largest element to its correct position with each pass
d. Using the divide-and-conquer approach

Question 5
Complete
Marked out of 1.00
Flag question

What is the primary benefit of using sorting algorithms in programming?

a. Makes code execution slower
b. Provides a basis for other algorithms to work efficiently
c. Decreases the efficiency of algorithms
d. Makes data harder to manage

Question 6
Complete
Marked out of 1.00
Flag question

Which of the following best describes the process of Merge Sort?

a. It builds a sorted array one element at a time
b. It repeatedly finds the minimum element and moves it to the sorted part of the list
c. It divides the list into two halves, sorts each half, and then merges them
d. It compares adjacent elements and swaps them if necessary

Question 7
Complete
Marked out of 1.00
Flag question

Which of the following is not an in-place sorting algorithm?

a. Quick sort
b. Selection sort
c. Heap sort
d. Merge sort

Question 8
Complete
Marked out of 1.00
Flag question

Which Python function would you use to sort a list in-place?

a. arrange()
b. order()
c. sort()
d. sorted()

Question 9

Complete

Marked out of 1.00

Flag question

What is one advantage of sorting a list before performing a search operation?

a. It has no effect on the search operation
b. It increases the number of comparisons needed
c. It makes the search operation slower
d. It allows for faster searching

Question 10

Complete

Marked out of 1.00

Flag question

What is a characteristic of the merge sort algorithm?

a. It is based on the divide-and-conquer approach
b. It sorts data using a single pass
c. It does not require recursion
d. It is less efficient than bubble sort

Question 11

Complete

Marked out of 1.00

Flag question

Which built-in Python function is used to sort data?

a. order()
b. sort()
c. sorted()
d. arrange()

Question 12

Complete

Marked out of 1.00

Flag question

Which sorting algorithm involves comparing elements and swapping adjacent items that are out

a. Binary Search
b. Merge Sort
c. Linear Search
d. Bubble Sort

Question 13

What is sorting in the context of computer science?

Complete
Marked out of 1.00
Flag question

a. Binary Search
b. Merge Sort
c. Linear Search
d. Bubble Sort

Question 13
Complete
Marked out of 1.00
Flag question

What is sorting in the context of computer science?

a. Deleting data from a list
b. Inserting data into a list
c. Arranging data in a particular format
d. Searching for data in a list

Question 14
Complete
Marked out of 1.00
Flag question

Which of the following is a key reason for the importance of sorting algorithms?

a. Sorting makes it harder to search for items
b. Sorting helps in finding duplicates quickly
c. Sorting decreases the efficiency of selection operations
d. Sorting is rarely used in programming

Question 15
Complete
Marked out of 1.00
Flag question

What type of problems can sorting help solve efficiently?

a. All of the above
b. Selection
c. Searching
d. Duplicates

ag question

Input: nums = [-4,-1,0,3,10]
Output: [0,1,9,16,100]
Explanation: After squaring, the array becomes [16,1,0,9,100].
After sorting, it becomes [0,1,9,16,100].

**Example 2:**

Input: nums = [-7,-3,2,3,11]
Output: [4,9,9,49,121]

**Constraints:**

- 1 <= nums.length <= $10^4$
- $-10^4$ <= nums[i] <= $10^4$
- nums is sorted in **non-decreasing** order.

**For example:**

| Test | Result |
| --- | --- |
| print(sortedSquares([-4,-1,0,3,10])) | [0, 1, 9, 16, 100] |

**Answer:** (penalty regime: 0 %)

Reset answer

```
def sortedSquares(n: list[int]) -> list[int]:
    a=[]
    for i in n :
        a.append(i**2)
    for i in range(0,len(a)) :
        for j in range(i+1,len(a)) :
            if a[i]>a[j] :
                a[i],a[j]=a[j],a[i]
    return a
```

**Example 3:**

```
Input:
9
-1 1 -6 4 5 -6 1 4 1
Output:
5 -1 4 4 -6 -6 1 1 1
```

**Constraints:**

- `1 <= nums.length <= 100`
- `-100 <= nums[i] <= 100`

**For example:**

| Input | Result |
|---|---|
| 6<br>1 1 2 2 2 3 | 3 1 1 2 2 2 |
| 5<br>2 3 1 3 2 | 1 3 3 2 2 |

**Answer:** (penalty regime: 0 %)

```
a=input()
b=list(map(int,input().split()))
c=set(b)
d={}
for i in c :
    co=b.count(i)
    d.update({i:co})
b.clear()
b=dict(sorted(d.items(),key=lambda item :(item[1], -item[0])))
p=[]
for i in b :
    count=b[i]
    for j in range(b[i]) :
        p.append(i)
print(*p)
```

Input: nums = [-1,-1]
Output: [0,0]

**Constraints:**

- $1 <= nums.length <= 10^5$
- $-10^4 <= nums[i] <= 10^4$

**For example:**

| Test | Result |
|---|---|
| print(countSmaller([5,2,6,1])) | [2, 1, 1, 0] |
| print(countSmaller([-1])) | [0] |

**Answer:** (penalty regime: 0 %)

Reset answer

```
def countSmaller(n:list[int]) -> list[int]:
    a=[]
    for i in range (0,len(n)) :
        count=0
        for j in range(i+1,len(n)) :
            if n[i]>n[j] :
                count+=1
        a.append(count)
    return a
```

**Constraints:**

- $0 <= arr.length <= 10^5$
- $-10^9 <= arr[i] <= 10^9$

**For example:**

| Test | Result |
| --- | --- |
| `print(arrayRankTransform([40,10,20,30]))` | `[4, 1, 2, 3]` |

**Answer:** (penalty regime: 0 %)

Reset answer

```
def arrayRankTransform(arr: list[int]) -> list[int]:
    a=list(set(arr))
    a.sort()
    b={}
    count=0
    for i in a :
        count+=1
        b.update({i:count})
    p=[]
    for i in arr:
        p.append(b[i])
    return p
```

| | Test | Expected |
| --- | --- | --- |
| ✔ | `print(arrayRankTransform([40,10,20,30]))` | `[4, 1, 2, 3]` |

ion 5

ct

1.00 out of

g question

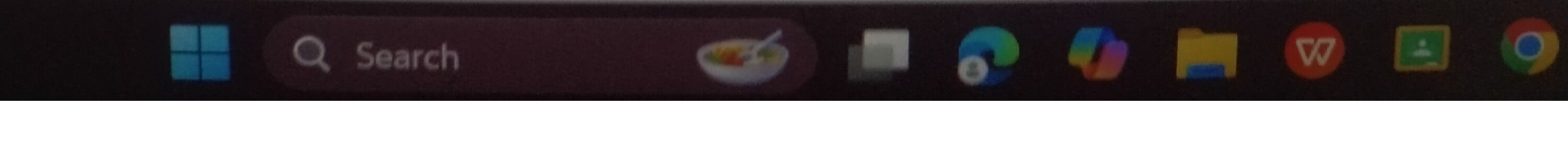
Write a Python program to sort a list of elements using the merge sort algorithm.

**For example:**

| Input | Result |
| --- | --- |
| 5<br>6 5 4 3 8 | 3 4 5 6 8 |

**Answer:** (penalty regime: 0 %)

```
'''def ms(n:list[int]) :
    if len(n)>1 :
        mid=len(n)//2
        l=n[:mid]
        r=n[mid:]
        ms(l)
        ms(r)
        n.clear()
        i=j=k=0
        while i<len(l) and j<len(r) :
            if l[i]<r[j] :
                n[k]=l[i]
                i+=1
            else :
                n[k]=r[j]
                j+=1
            k+=1
        while i<len(l):
            n[k]=l[i]
            i+=1
            k+=1
        while j<len(r) :
            n[k]=r[j]
            j+=1
            k+=1
    return n
'''
a=int(input())
b=list(map(int,input().split()))
b.sort()
print(*b)
```



- **Explanation:** The distinct elements in the list are [1, 2]. There is no 3rd maximum valu

**For example:**

| Input | Result |
|---|---|
| 5<br>3 1 5 4 2<br>2 | 4 |
| 6<br>7 7 7 7 7 7<br>1 | 7 |
| 10<br>2 1 2 1 2 1 2 1 2 1<br>3 | -1 |

**Answer:** (penalty regime: 0 %)

```
a=input()
b=list(set(map(int,input().split())))
c=int(input())
count=0
if c>len(b) :
    print(-1)
else :
    for i in range(len(b)) :
        for j in range(len(b)) :
            if b[i]>b[j] :
                b[i],b[j]=b[j],b[i]
    for i in range(c) :
        count=b[i]
    print(count)
```

| | Input | Expected | Got | |
|---|---|---|---|---|

Question 7

Correct

Mark 1.00 out of 1.00

Flag question

The problem is that we want to reverse a array in O(N) linear time complexity and we want the algorithm to be in-place as well!

For example: input is [1,2,3,4,5] then the output is [5,4,3,2,1]

Input

5

1 2 3 4 5

Output

5 4 3 2 1

**For example:**

| Input | Result |
|---|---|
| 5<br>1 2 3 4 5 | 5 4 3 2 1 |

**Answer:** (penalty regime: 0 %)

```
a=input()
b=list(map(int,input().split()))
print(*b[::-1])
```

| | Input | Expected | Got |
|---|---|---|---|

**For example:**

| Input | Result |
|---|---|
| 3<br>3 2 1 | List is sorted in 3 swaps.<br>First Element: 1<br>Last Element: 3 |
| 5<br>1 9 2 8 4 | List is sorted in 4 swaps.<br>First Element: 1<br>Last Element: 9 |

**Answer:** (penalty regime: 0 %)

```
a=int(input())
count=0
b=list(map(int,input().split()))
for i in range(len(b)) :
    for j in range(i+1,len(b)) :
        if b[i]>b[j] :
            b[i],b[j]=b[j],b[i]
            count+=1
print("List is sorted in %d swaps."%count)
print("First Element: %d"%b[0])
print("Last Element: %d"%b[-1])
```

| | Input | Expected | Got | |
|---|---|---|---|---|
| ✔ | 3<br>3 2 1 | List is sorted in 3 swaps.<br>First Element: 1<br>Last Element: 3 | List is sorted in 3 swaps.<br>First Element: 1<br>Last Element: 3 | ✔ |
| ✔ | 5<br>1 9 2 8 4 | List is sorted in 4 swaps.<br>First Element: 1 | List is sorted in 4 swaps.<br>First Element: 1 | ✔ |

**For example:**

| Input | Result |
| --- | --- |
| 6<br>3 4 8 7 1 2 | 1 2 3 4 7 8 |
| 5<br>4 5 2 3 1 | 1 2 3 4 5 |

**Answer:** (penalty regime: 0 %)

```
a=input()
b=list(map(int,input().split()))
for i in range(len(b)) :
    for j in range(len(b)):
        if b[i]<b[j] :
            b[i],b[j]=b[j],b[i]
print(*b)
```

| | Input | Expected | Got | |
| --- | --- | --- | --- | --- |
| ✔ | 6<br>3 4 8 7 1 2 | 1 2 3 4 7 8 | 1 2 3 4 7 8 | ✔ |
| ✔ | 6<br>9 18 1 3 4 6 | 1 3 4 6 9 18 | 1 3 4 6 9 18 | ✔ |
| ✔ | 5 | 1 2 3 4 5 | 1 2 3 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<br>4 2<br>5 2 |

**Answer:** (penalty regime: 0 %)

```
a=list(map(int,input().split()))
for i in range(len(a)) :
    for j in range(len(a)) :
        if a[i]>a[j] :
            a[i],a[j]=a[j],a[i]
b=set(a)
for i in b :
     print(i,a.count(i))
```