Quiz navigation

1 ✓ 2 ✓ 3 ✓ 4 ✓ 5 ✓

Show one page at a time

Finish review

Started on	Tuesday, 28 May 2024, 1:48 PM
State	Finished
Completed on	Tuesday, 28 May 2024, 1:56 PM
Time taken	7 mins 10 secs
Marks	5.00/5.00
Grade	**100.00** out of 100.00

Question **1**
Correct
Mark 1.00 out of 1.00
Flag question

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

Answer: (penalty regime: 0 %)

```
def find_peaks(n, A):
    peaks = []
    for i in range(n):

        if (i == 0 and A[i] >= A[i + 1]) or \
           (i == n - 1 and A[i] >= A[i - 1]) or \
           (0 < i < n - 1 and A[i] >= A[i - 1] and A[i] >= A[i + 1]):
            peaks.append(A[i])
    return peaks


n = int(input())
A = list(map(int, input().split()))


peaks = find_peaks(n, A)
print(" ".join(map(str, peaks)))
```

	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.

Question **2**
Correct
Mark 1.00 out of 1.00
Flag question

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

Answer: (penalty regime: 0 %)

```
def bubble_sort(arr):
    n = len(arr)
    num_swaps = 0

    for i in range(n):

        swapped = False

        for j in range(0, n - i - 1):
            if arr[j] > arr[j + 1]:

                arr[j], arr[j + 1] = arr[j + 1], arr[j]
                num_swaps += 1
                swapped = True


        if not swapped:
            break

    return arr, num_swaps


n = int(input())
a = list(map(int, input().split()))


sorted_list, num_swaps = bubble_sort(a)


print(f"List is sorted in {num_swaps} swaps.")
print(f"First Element: {sorted_list[0]}")
print(f"Last Element: {sorted_list[-1]}")
```

	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.

Question **3**
Correct
Mark 1.00 out of 1.00
Flag question

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

Answer: (penalty regime: 0 %)

```
n = int(input())
arr = list(map(int, input().split()))
def merge_sort(arr):
    if len(arr) <= 1:
        return arr
    mid = len(arr) // 2
    left_half = merge_sort(arr[:mid])
    right_half = merge_sort(arr[mid:])
    return sorted(left_half + right_half)
print(*merge_sort(arr))
```

	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 57 70	✓
✓	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.

Question **4**
Correct
Mark 1.00 out of 1.00
Flag question

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

Answer: (penalty regime: 0 %)

```
A = list(map(int, input().split()))
for B in sorted(set(A)):
    print(B, A.count(B))
```

	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.

Question **5**
Correct
Mark 1.00 out of 1.00
Flag question

Bubble Sort is the simplest sorting algorithm that works by repeatedly swapping the adjacent elements if they are in wrong order. You read an list of numbers. You need to arrange the elements in ascending order and print the result. The sorting should be done using bubble sort.

Input Format: The first line reads the number of elements in the array. The second line reads the array elements one by one.

Output Format: The output should be a sorted list.

For example:

Input	Result
6 3 4 8 7 1 2	1 2 3 4 7 8
5 4 5 2 3 1	1 2 3 4 5

Answer: (penalty regime: 0 %)

```
def bubble_sort(arr):
    n = len(arr)
    for i in range(n):
        for j in range(0, n-i-1):
            if arr[j] > arr[j+1]:
                arr[j], arr[j+1] = arr[j+1], arr[j]
n = int(input())
arr = list(map(int, input().split()))
bubble_sort(arr)
print(*arr)

```

	Input	Expected	Got	
✓	6 3 4 8 7 1 2	1 2 3 4 7 8	1 2 3 4 7 8	✓
✓	6 9 18 1 3 4 6	1 3 4 6 9 18	1 3 4 6 9 18	✓
✓	5 4 5 2 3 1	1 2 3 4 5	1 2 3 4 5	✓

Passed all tests! ✓

Correct
Marks for this submission: 1.00/1.00.

Finish review

◄ Week10_MCQ Jump to... Sorting ►