

[Dashboard](#) / [My courses](#) / [PSP/PUP](#) / [Searching techniques: Linear and Binary](#) / [Week10 Coding](#)

Started on	Saturday, 25 May 2024, 10:41 AM
State	Finished
Completed on	Saturday, 25 May 2024, 10:50 AM
Time taken	8 mins 42 secs
Marks	5.00/5.00
Grade	100.00 out of 100.00

Question 1

Correct

Mark 1.00 out of 1.00

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 %)

```

1 | n = int(input())
2 | arr = list(map(int, input().split()))
3 |
4 |
5 | for i in range(n):
6 |     for j in range(0, n-i-1):
7 |         if arr[j] > arr[j+1]:
8 |             arr[j], arr[j+1] = arr[j+1], arr[j]
9 |
10 |
11 | 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.

Question 2

Correct

Mark 1.00 out of 1.00

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

$A[i-1] \leq A[i] \geq A[i+1]$ for middle elements. $[0 < i < n-1]$

$A[i-1] \leq A[i]$ for last element $[i=n-1]$

$A[i] \geq A[i+1]$ 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 %)

```

1 | n = int(input())
2 | arr = list(map(int, input().split()))
3 |
4 | p = []
5 | if arr[0] >= arr[1]:
6 |     p.append(arr[0])
7 | for i in range(1, n - 1):
8 |     if arr[i - 1] <= arr[i] >= arr[i + 1]:
9 |         p.append(arr[i])
10 | if arr[-1] >= arr[-2]:
11 |     p.append(arr[-1])
12 |
13 | print(*p)
```

	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 3

Correct

Mark 1.00 out of 1.00

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

Answer: (penalty regime: 0 %)

```

1 n=input()
2 k=(input())
3 if k in n:
4     print(True)
5 else:
6     print(False)

```

	Input	Expected	Got	
✓	1,2,3,5,8 6	False	False	✓
✓	3,5,9,45,42 42	True	True	✓
✓	52,45,89,43,11 11	True	True	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Question 4

Correct

Mark 1.00 out of 1.00

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 %)

```

1  n = int(input())
2  arr = list(map(int, input().split()))
3
4
5  stack = [(0, n)]
6  while stack:
7      start, end = stack.pop()
8      if end - start > 1:
9          mid = (start + end) // 2
10
11         stack.extend([(start, mid), (mid, end)])
12         arr[start:end] = sorted(arr[start:end])
13
14
15 print(*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 5

Correct

Mark 1.00 out of 1.00

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

- $2 \leq n \leq 600$
- $1 \leq a[i] \leq 2 \times 10^6$.

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 %)

```

1 n = int(input())
2 a = list(map(int, input().split()))
3
4 swaps = 0
5 for i in range(n):
6     for j in range(n-1):
7         if a[j] > a[j+1]:
8             a[j], a[j+1] = a[j+1], a[j]
9             swaps += 1
10
11 print(f"List is sorted in {swaps} swaps.")

```

```
12 print(f"First Element: {a[0]}")
13 print(f"Last Element: {a[-1]}")
14
```

	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.

[◀ Week10_MCQ](#)

Jump to...