Dashboard / My courses / PSPP/PUP / Searching techniques: Linear and Binary / Week10_Coding

Started on	Friday, 24 May 2024, 9:51 AM
State	Finished
Completed on	Sunday, 26 May 2024, 7:01 PM
Time taken	2 days 9 hours
Marks	5.00/5.00
Grade	**100.00** out of 100.00

Question **1**

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 %)

```python
def BinarySearch(a,key):
    a = sorted(a)
    first = 0
    last = len(a) - 1
    while first <= last:
        mid = (first+last)//2
        if a[mid] == key:
            return mid
        elif a[mid] < key:
            first = mid + 1
        else:
            last = mid - 1
    return -1

a = list(map(int, input().split(',')))
key = int(input())
pos = BinarySearch(a, key)

if pos != -1:
    print("True")
else:
    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 **2**

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 %)

```python
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]
    return arr
n = int(input())
array = list(map(int, input().split()))
sorted_array = bubble_sort(array)
print(*sorted_array)
```

	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 **3**

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] <= 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 %)

```python
def find_peak_elements(arr):
    n = len(arr)
    peak_elements = []
    if n == 1 or arr[0] >= arr[1]:
        peak_elements.append(arr[0])
    for i in range(1, n-1):
        if arr[i] >= arr[i-1] and arr[i] >= arr[i+1]:
            peak_elements.append(arr[i])
    if n > 1 and arr[-1] >= arr[-2]:
        peak_elements.append(arr[-1])
    return peak_elements

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

peak_elements = find_peak_elements(arr)
print(*peak_elements)
```

	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 **4**

Correct

Mark 1.00 out of 1.00

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 %)

```python
def find_frequency(arr):
    freq = {}
    for num in arr:
        if num in freq:
            freq[num] += 1
        else:
            freq[num] = 1

    sorted_freq = sorted(freq.items(), key=lambda x: x[0])
    return sorted_freq

arr = list(map(int, input().split()))

sorted_freq = find_frequency(arr)
for num, count in sorted_freq:
    print(num, count)
```

	Input	Expected	Got	
✓	4 3 5 3 4 5	3 2 4 2 5 2	3 2 4 2 5 2	✓

	Input	Expected	Got	
✓	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

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 %)

```python
def merge_sort(arr):
    if len(arr) <= 1:
        return arr

    mid = len(arr) // 2
    left_half = arr[:mid]
    right_half = arr[mid:]

    left_half = merge_sort(left_half)
    right_half = merge_sort(right_half)

    return merge(left_half, right_half)

def merge(left, right):
    result = []
    left_index = 0
    right_index = 0

    while left_index < len(left) and right_index < len(right):
        if left[left_index] <= right[right_index]:
            result.append(left[left_index])
            left_index += 1
        else:
            result.append(right[right_index])
            right_index += 1

    while left_index < len(left):
        result.append(left[left_index])
        left_index += 1

    while right_index < len(right):
        result.append(right[right_index])
        right_index += 1

    return result

n = int(input())
arr = list(map(int, input().split()))
sorted_arr = merge_sort(arr)
print(*sorted_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	✓

	Input	Expected	Got	
✓	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.

◄ Week10_MCQ

Jump to...

Sorting ►