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

Started on	Saturday, 15 June 2024, 9:16 PM
State	Finished
Completed on	Wednesday, 19 June 2024, 12:52 AM
Time taken	3 days 3 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 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 %)

```

a=int(input())
b = input().split()
b = [int(i) for i in b]
b.sort()
for i in b:
    print(i,end=" ")
```

	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 **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):
        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]
                swapped = True
        if not swapped:
            break
    return arr
n = int(input())
arr = list(map(int, input().split()))
sorted_arr = bubble_sort(arr)
print(" ".join(map(str, sorted_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 **3**

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 %)

```
n=list(map(int,input().split()))
freq={}
for num in n:
    freq[num]=freq.get(num,0)+1
sortf=sorted(freq.items())
for num,freqs in sortf:
    print(num,freqs)
```

	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	✓

	Input	Expected	Got	
✓	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 **4**

Correct

Mark 1.00 out of 1.00

An list contains N numbers and you want to determine whether two of the numbers sum to a given number K. For example, if the input is 8, 4, 1, 6 and K is 10, the answer is yes (4 and 6). A number may be used twice.

Input Format

The first line contains a single integer n , the length of list

The second line contains n space-separated integers, list[i].

The third line contains integer k.

Output Format

Print Yes or No.

Sample Input

7

0 1 2 4 6 5 3

1

Sample Output

Yes

For example:

Input	Result

5
8 9 12 15 3
11
	Yes

6
2 9 21 32 43 43 1
4
	No

Answer: (penalty regime: 0 %)

```
def two_sum(li,k):
    for i in range(len(li)):
        for j in range(i+1,len(li)):
            if li[i] +li[j] == k:
                return "Yes"
    return "No"

n = int(input())
li = input().split()
li = [int(x) for x in li]
k = int(input())
print(two_sum(li,k))
```

	Input	Expected	Got	
✓	5 8 9 12 15 3 11	Yes	Yes	✓
✓	6 2 9 21 32 43 43 1 4	No	No	✓
✓	6 13 42 31 4 8 9 17	Yes	Yes	✓

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<=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):
    num_swaps=0
    n=len(arr)
    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 num_swaps
n=int(input())
arr=list(map(int,input().split()))
num_swaps=bubble_sort(arr)
print("List is sorted in", num_swaps,"swaps.")
print("First Element:",arr[0])
print("Last Element:",arr[-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.

◄ Week10_MCQ

Jump to...

Sorting ►