Started on	Friday, 7 February 2025, 3:17 PM
State	Finished
Completed on	Friday, 7 February 2025, 3:21 PM
Time taken	4 mins 15 secs
Grade	**100.00** out of 100.00

Question **1**

Correct

Mark 20.00 out of 20.00

Write a python program for a search function with parameter list name and the value to be searched on the given list of float values.

For example:

Test	Input	Result
search(List, n)	5 3.2 6.1 4.5 6.2 8.5 3.2	3.2 Found
search(List, n)	4 3.2 1.5 6.4 7.8 6.1	6.1 Not Found

Answer: (penalty regime: 0 %)

```
List=[]
s=int(input())
for i in range(s):
    List.append(input())
n=input()
def search(List,n):
    for i in List:
        if i==n:
            print(n,"Found")
            break;
    else:
        print(n,"Not Found")
```

	Test	Input	Expected	Got	
✔	search(List, n)	5 3.2 6.1 4.5 6.2 8.5 3.2	3.2 Found	3.2 Found	✔

	Test	Input	Expected	Got	
✔	search(List, n)	4 3.2 1.5 6.4 7.8 6.1	6.1 Not Found	6.1 Not Found	✔
✔	search(List, n)	7 2.1 3.2 6.5 4.1 5.2 7.1 8.2 9.3	9.3 Not Found	9.3 Not Found	✔

Passed all tests! ✔

Correct

Marks for this submission: 20.00/20.00.

Question **2**

Correct

Mark 20.00 out of 20.00

Write a python program to implement quick sort on the given float array values.

For example:

Input	Result
5 6.9 8.3 2.1 1.5 6.4	left: [] right: [] left: [] right: [] left: [1.5] right: [6.4] left: [] right: [] left: [1.5, 2.1, 6.4] right: [8.3] [1.5, 2.1, 6.4, 6.9, 8.3]
6 3.1 2.4 5.6 4.3 6.2 7.8	left: [] right: [] left: [] right: [] left: [] right: [] left: [] right: [7.8] left: [4.3] right: [6.2, 7.8] left: [2.4] right: [4.3, 5.6, 6.2, 7.8] [2.4, 3.1, 4.3, 5.6, 6.2, 7.8]

Answer: (penalty regime: 0 %)

```
def quickSort(arr):
    if arr==[]:
        return arr
    pivot=arr[0:1]
    left=quickSort([x for x in arr[1:] if x<pivot[0]])
    right=quickSort([x for x in arr[1:] if x>=pivot[0]])
    print("left: ",left)
    print("right: ",right)
    return left+pivot+right

l=[float(input()) for i in range(int(input()))]
s=quickSort(l)
print(s)
```

	Input	Expected	Got	

	Input	Expected	Got	
✔	5 6.9 8.3 2.1 1.5 6.4	left: [] right: [] left: [] right: [] left: [1.5] right: [6.4] left: [] right: [] left: [1.5, 2.1, 6.4] right: [8.3] [1.5, 2.1, 6.4, 6.9, 8.3]	left: [] right: [] left: [] right: [] left: [1.5] right: [6.4] left: [] right: [] left: [1.5, 2.1, 6.4] right: [8.3] [1.5, 2.1, 6.4, 6.9, 8.3]	✔
✔	6 3.1 2.4 5.6 4.3 6.2 7.8	left: [] right: [] left: [] right: [] left: [] right: [] left: [] right: [7.8] left: [4.3] right: [6.2, 7.8] left: [2.4] right: [4.3, 5.6, 6.2, 7.8] [2.4, 3.1, 4.3, 5.6, 6.2, 7.8]	left: [] right: [] left: [] right: [] left: [] right: [] left: [] right: [7.8] left: [4.3] right: [6.2, 7.8] left: [2.4] right: [4.3, 5.6, 6.2, 7.8] [2.4, 3.1, 4.3, 5.6, 6.2, 7.8]	✔
✔	8 1.2 1.3 4.2 5.3 6.4 7.3 6.8 9.2	left: [] right: [] left: [] right: [] left: [6.8] right: [9.2] left: [] right: [6.8, 7.3, 9.2] left: [] right: [6.4, 6.8, 7.3, 9.2] left: [] right: [5.3, 6.4, 6.8, 7.3, 9.2] left: [] right: [4.2, 5.3, 6.4, 6.8, 7.3, 9.2] left: [] right: [1.3, 4.2, 5.3, 6.4, 6.8, 7.3, 9.2] [1.2, 1.3, 4.2, 5.3, 6.4, 6.8, 7.3, 9.2]	left: [] right: [] left: [] right: [] left: [6.8] right: [9.2] left: [] right: [6.8, 7.3, 9.2] left: [] right: [6.4, 6.8, 7.3, 9.2] left: [] right: [5.3, 6.4, 6.8, 7.3, 9.2] left: [] right: [4.2, 5.3, 6.4, 6.8, 7.3, 9.2] left: [] right: [1.3, 4.2, 5.3, 6.4, 6.8, 7.3, 9.2] [1.2, 1.3, 4.2, 5.3, 6.4, 6.8, 7.3, 9.2]	✔

Passed all tests! ✔

Correct

Marks for this submission: 20.00/20.00.

Question **3**

Correct

Mark 20.00 out of 20.00

Write a Python Program to print the fibonacci series upto n_terms using Recursion.

For example:

Input	Result
10	Fibonacci series: 0 1 1 2 3 5 8 13 21 34
5	Fibonacci series: 0 1 1 2 3
7	Fibonacci series: 0 1 1 2 3 5 8

Answer: (penalty regime: 0 %)

```
def fibonacci(n):
    if n <= 0:
        return 0
    elif n == 1:
        return 1;
    else:
        return fibonacci(n-1) + fibonacci(n-2)

def print_fibonacci_series(n_terms):
    print("Fibonacci series:")
    for i in range(n_terms):
        print(fibonacci(i))

n_terms = int(input())
print_fibonacci_series(n_terms)
```

	Input	Expected	Got	

	Input	Expected	Got	
✔	10	Fibonacci series: 0 1 1 2 3 5 8 13 21 34	Fibonacci series: 0 1 1 2 3 5 8 13 21 34	✔
✔	5	Fibonacci series: 0 1 1 2 3	Fibonacci series: 0 1 1 2 3	✔
✔	7	Fibonacci series: 0 1 1 2 3 5 8	Fibonacci series: 0 1 1 2 3 5 8	✔
✔	9	Fibonacci series: 0 1 1 2 3 5 8 13 21	Fibonacci series: 0 1 1 2 3 5 8 13 21	✔
✔	11	Fibonacci series: 0 1 1 2 3 5 8 13 21 34 55	Fibonacci series: 0 1 1 2 3 5 8 13 21 34 55	✔

Passed all tests! ✔

Correct

Marks for this submission: 20.00/20.00.

Question **4**

Correct

Mark 20.00 out of 20.00

Write a python program to implement merge sort using iterative approach on the given list of values.

For example:

Test	Input	Result
Merge_Sort(S)	6 4 2 3 1 6 5	The Original array is: [4, 2, 3, 1, 6, 5] Array after sorting is: [1, 2, 3, 4, 5, 6]
Merge_Sort(S)	5 2 6 4 3 1	The Original array is: [2, 6, 4, 3, 1] Array after sorting is: [1, 2, 3, 4, 6]

Answer: (penalty regime: 0 %)

```
def Merge_Sort(S):
    n = len(S)
    current_size = 1

    while current_size < n:
        left = 0
        while left < n - 1:
            mid = min(left + current_size - 1, n - 1)
            right = min(left + 2 * current_size - 1, n - 1)

            merge(S, left, mid, right)
            left += 2 * current_size

        current_size *= 2


def merge(S, left, mid, right):
    n1 = mid - left + 1
```

	Test	Input	Expected	Got	
✔	Merge_Sort(S)	6 4 2 3 1 6 5	The Original array is: [4, 2, 3, 1, 6, 5] Array after sorting is: [1, 2, 3, 4, 5, 6]	The Original array is: [4, 2, 3, 1, 6, 5] Array after sorting is: [1, 2, 3, 4, 5, 6]	✔
✔	Merge_Sort(S)	5 2 6 4 3 1	The Original array is: [2, 6, 4, 3, 1] Array after sorting is: [1, 2, 3, 4, 6]	The Original array is: [2, 6, 4, 3, 1] Array after sorting is: [1, 2, 3, 4, 6]	✔

	Test	Input	Expected	Got	
✔	Merge_Sort(S)	4 3 5 6 1	The Original array is: [3, 5, 6, 1] Array after sorting is: [1, 3, 5, 6]	The Original array is: [3, 5, 6, 1] Array after sorting is: [1, 3, 5, 6]	✔

Passed all tests! ✔

Correct

Marks for this submission: 20.00/20.00.

Question **5**

Correct

Mark 20.00 out of 20.00

Write a python program to implement binary search on the given list of float values using iterative method

For example:

Test	Input	Result
`binarySearchAppr(arr, 0, len(arr)-1, x)`	5 3.2 6.1 4.5 9.6 8.3 6.1	Element is present at index 2
`binarySearchAppr(arr, 0, len(arr)-1, x)`	6 3.1 2.3 5.1 4.6 3.2 9.5 4.6	Element is present at index 3

Answer: (penalty regime: 0 %)

```
def binarySearchAppr (arr, start, end, x):
    if end >= start:
       mid = (start +end)//2
       if arr[mid] == x:
           return mid
       elif arr[mid] > x:
           return binarySearchAppr(arr, start, mid-1, x)
       else:
          return binarySearchAppr(arr,mid+1,end,x)
    else:
      return -1
arr=[]
n=int(input())
for i in range(n):
    arr.append(input())
arr = sorted(arr)
x =input()
result = binarySearchAppr(arr,0,len(arr)-1,x)
```

	Test	Input	Expected	Got	
✔	`binarySearchAppr(arr, 0, len(arr)-1, x)`	5 3.2 6.1 4.5 9.6 8.3 6.1	Element is present at index 2	Element is present at index 2	✔

	Test	Input	Expected	Got	
✔	binarySearchAppr(arr, 0, len(arr)-1, x)	6 3.1 2.3 5.1 4.6 3.2 9.5 4.6	Element is present at index 3	Element is present at index 3	✔
✔	binarySearchAppr(arr, 0, len(arr)-1, x)	8 2.1 6.3 5.2 4.2 9.3 6.7 5.6 9.8 7.2	Element is not present in array	Element is not present in array	✔

Passed all tests! ✔

Correct

Marks for this submission: 20.00/20.00.