Started on	Thursday, 20 March 2025, 8:17 AM
State	Finished
Completed on	Thursday, 20 March 2025, 9:06 AM
Time taken	48 mins 48 secs
Grade	**80.00** 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 %)

```python
global key
def search(List,n):
    for i in range(n):
        if(List[i]==key):
            return i
    else:
        return -1
List=[]
n=int(input())
for i in range(n):
    x=float(input())
    List.append(x)
key=float(input())
res=search(List,n)
if(res!=-1):
    print(f"{key} Found")
else:
    print(f"{key} 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 linear search on the given tuple of float values.

note: As the tuple is immutable convert the list to tuple to perform search

For example:

Input	Result
5 3.2 1.5 6.4 7.8 9.5 6.4	Tuple: 6.4 found
6 3.2 1.2 3.4 5.3 6.2 6.8 6.2	Tuple: 6.2 found

Answer: (penalty regime: 0 %)

```python
def search(List,key,n):
    for i in range(n):
        if(List[i]==key):
            return i
    else:
        return -1
List=[]
n=int(input())
for i in range(n):
    x=float(input())
    List.append(x)
key=float(input())
res=search(List,key,n)
if(res!=-1):
    print(f"Tuple: {key} found")
else:
    print(f"Tuple: {key} not found")
```

	Input	Expected	Got	
✓	5 3.2 1.5 6.4 7.8 9.5 6.4	Tuple: 6.4 found	Tuple: 6.4 found	✓

	Input	Expected	Got	
✓	6 3.2 1.2 3.4 5.3 6.2 6.8 6.2	Tuple: 6.2 found	Tuple: 6.2 found	✓
✓	4 2.1 3.2 6.5 4.5 3.5	Tuple: 3.5 not found	Tuple: 3.5 not found	✓

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 fib(n):
    f=0
    s=1
    print("Fibonacci series:")
    for i in range(n):
        print(f)
        t=f+s
        f=s
        s=t
n=int(input())
fib(n)
```

	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 %)

```python
def Merge_Sort(S):
    size=len(S)
    if(size>1):
        mid=size//2
        l=S[:mid]
        r=S[mid:]
        Merge_Sort(l)
        Merge_Sort(r)
        i=0
        j=0
        k=0
        l_s=len(l)
        r_s=len(r)
        while(i<l_s and j<r_s):
            if(l[i]<r[j]):
                S[k]=l[i]
                i+=1
            else:
                S[k]=r[j]
                j+=1
            k+=1
        while(i<l_s):
```

	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]`
					✓

	Test	Input	Expected	Got	

✓	`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]`
					✓

✓	`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**

Incorrect

Mark 0.00 out of 20.00

Write a python program to implement the quick sort using recursion on the given list of float values.

For example:

Input	Result
5 6.3 1.2 4.6 5.8 9.7	pivot: 9.7 pivot: 5.8 pivot: 4.6 [1.2, 4.6, 5.8, 6.3, 9.7]
6 2.3 7.8 9.5 4.2 3.6 5.4	pivot: 5.4 pivot: 3.6 pivot: 7.8 [2.3, 3.6, 4.2, 5.4, 7.8, 9.5]

Answer: (penalty regime: 0 %)

```python
def part(a,l,h):
    p=a[h]
    print("pivot: ",p)
    i=l
    j=h-1
    while(i<=j):
        while(i<=j and a[i]<p):
            i+=1
        while(i<=j and a[j]>=p):
            j-=1
        if(i<j):
            a[i],a[j]=a[j],a[i]
    a[i],a[h]=a[h],a[i]
    return i
def qs(a,l,h):
    if(l<h):
        p=part(a,l,h)
        qs(a,l,p-1)
        qs(a,p+1,h)
a=[]
n=int(input())
for i in range(n):
```

	Input	Expected	Got	
✗	5 6.3 1.2 4.6 5.8 9.7	pivot: 9.7 pivot: 5.8 pivot: 4.6 [1.2, 4.6, 5.8, 6.3, 9.7]	pivot: 9.7 pivot: 5.8 pivot: 1.2 [1.2, 4.6, 5.8, 6.3, 9.7]	✗

Some hidden test cases failed, too.

Your code must pass all tests to earn any marks. Try again.

Show differences

Incorrect

Marks for this submission: 0.00/20.00.