Started on	Saturday, 26 April 2025, 9:00 AM
State	Finished
Completed on	Saturday, 26 April 2025, 10:06 AM
Time taken	1 hour 6 mins
Grade	**100.00** out of 100.00

Question **1**

Correct

Mark 20.00 out of 20.00

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

For example:

Test	Input	Result
binarySearchAppr(arr, 0, len(arr)-1, x)	5 one two three four five two	Element is present at index 4
binarySearchAppr(arr, 0, len(arr)-1, x)	6 one three five seven nine eleven thirteen	Element is not present in array

Answer: (penalty regime: 0 %)

```
def binarySearchAppr(arr, low, high, x):
    arr.sort()
    while low <= high:
        mid = (low + high) // 2
        if arr[mid] == x:
            return f"Element is present at index {mid}"
        elif arr[mid] < x:
            low = mid + 1
        else:
            high = mid - 1
    return "Element is not present in array"

n = int(input())
arr = [input() for _ in range(n)]
x = input()
print(binarySearchAppr(arr, 0, len(arr)-1, x))

```

	Test	Input	Expected	Got	
✔	binarySearchAppr(arr, 0, len(arr)-1, x)	5 one two three four five two	Element is present at index 4	Element is present at index 4	✔
✔	binarySearchAppr(arr, 0, len(arr)-1, x)	6 one three five seven nine eleven thirteen	Element is not present in array	Element is not present in array	✔

	Test	Input	Expected	Got	

✔	`binarySearchAppr(arr, 0, len(arr)-1, x)`	`4`
		`two`
		`four`
		`six`
		`eight`
		`six`
			`Element is present at index 2`	`Element is present at index 2`	✔

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 print factorial of a number recursively.

For example:

Input	Result
5	Factorial of number 5 = 120
6	Factorial of number 6 = 720

Answer: (penalty regime: 0 %)

```
def factorial(n):
    if n == 0 or n == 1:
        return 1
    return n * factorial(n - 1)
n = int(input())
print(f"Factorial of number {n} = {factorial(n)}")

```

	Input	Expected	Got	
✔	5	Factorial of number 5 = 120	Factorial of number 5 = 120	✔
✔	6	Factorial of number 6 = 720	Factorial of number 6 = 720	✔
✔	7	Factorial of number 7 = 5040	Factorial of number 7 = 5040	✔
✔	8	Factorial of number 8 = 40320	Factorial of number 8 = 40320	✔

Passed all tests! ✔

Marks for this submission: 20.00/20.00.

Question **3**

Correct

Mark 20.00 out of 20.00

Write a python program to implement merge sort without using recursive function on the given list of values.

For example:

Input	Result
7 33 42 9 37 8 47 5	left: [33] Right: [42] left: [9] Right: [37] left: [8] Right: [47] left: [5] Right: [] left: [33, 42] Right: [9, 37] left: [8, 47] Right: [5] left: [9, 33, 37, 42] Right: [5, 8, 47] [5, 8, 9, 33, 37, 42, 47]
6 10 3 5 61 74 92	left: [10] Right: [3] left: [5] Right: [61] left: [74] Right: [92] left: [3, 10] Right: [5, 61] left: [74, 92] Right: [] left: [3, 5, 10, 61] Right: [74, 92] [3, 5, 10, 61, 74, 92]

Answer: (penalty regime: 0 %)

```
def merge_sort_iterative(arr):
    width = 1
    n = len(arr)
    while width < n:
        l = 0
        while l < n:
            r = min(l + (width * 2 - 1), n - 1)
            m = min(l + width - 1, n - 1)
            left = arr[l:m+1]
            right = arr[m+1:r+1]
            print(f"left:  {left}")
            print(f"Right:  {right}")
            i = j = 0
            k = l
            while i < len(left) and j < len(right):
                if left[i] <= right[j]:
                    arr[k] = left[i]
                    i += 1
                else:
                    arr[k] = right[j]
                    j += 1
                k += 1
```

	Input	Expected	Got	

	Input	Expected	Got	
✔	7 33 42 9 37 8 47 5	left: [33] Right: [42] left: [9] Right: [37] left: [8] Right: [47] left: [5] Right: [] left: [33, 42] Right: [9, 37] left: [8, 47] Right: [5] left: [9, 33, 37, 42] Right: [5, 8, 47] [5, 8, 9, 33, 37, 42, 47]	left: [33] Right: [42] left: [9] Right: [37] left: [8] Right: [47] left: [5] Right: [] left: [33, 42] Right: [9, 37] left: [8, 47] Right: [5] left: [9, 33, 37, 42] Right: [5, 8, 47] [5, 8, 9, 33, 37, 42, 47]	✔
✔	6 10 3 5 61 74 92	left: [10] Right: [3] left: [5] Right: [61] left: [74] Right: [92] left: [3, 10] Right: [5, 61] left: [74, 92] Right: [] left: [3, 5, 10, 61] Right: [74, 92] [3, 5, 10, 61, 74, 92]	left: [10] Right: [3] left: [5] Right: [61] left: [74] Right: [92] left: [3, 10] Right: [5, 61] left: [74, 92] Right: [] left: [3, 5, 10, 61] Right: [74, 92] [3, 5, 10, 61, 74, 92]	✔
✔	5 4 12 6 98 3	left: [4] Right: [12] left: [6] Right: [98] left: [3] Right: [] left: [4, 12] Right: [6, 98] left: [3] Right: [] left: [4, 6, 12, 98] Right: [3] [3, 4, 6, 12, 98]	left: [4] Right: [12] left: [6] Right: [98] left: [3] Right: [] left: [4, 12] Right: [6, 98] left: [3] Right: [] left: [4, 6, 12, 98] Right: [3] [3, 4, 6, 12, 98]	✔

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 for a search function with parameter list name and the value to be searched on the given list of int values.

For example:

Test	Input	Result
search(List, n)	5 3 4 5 6 7 4	Found
search(List, n)	6 20 34 56 87 96 51 87	Found

Answer: (penalty regime: 0 %)

```
global key
def search(list, n):
    for i in range(0,n):
        if(list[i]==key):
            return i
    return -1
List = []
n=int(input())
for i in range(0,n):
    temp=input()
    List.append(temp)
key=input()
res=search(List, n)
if(res==-1):
    print("Not Found")
else:
    print("Found")

```

	Test	Input	Expected	Got	
✔	search(List, n)	5 3 4 5 6 7 4	Found	Found	✔

	Test	Input	Expected	Got	
✔	search(List, n)	6 20 34 56 87 96 51 87	Found	Found	✔
✔	search(List, n)	4 30 10 20 50 60	Not Found	Not Found	✔

Passed all tests! ✔

Marks for this submission: 20.00/20.00.

Question **5**

Correct

Mark 20.00 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 %)

```
def partition(l, r, nums):
    pivot, ptr = nums[r], l
    for i in range(l, r):
        if nums[i] <= pivot:
            nums[i], nums[ptr] = nums[ptr], nums[i]
            ptr += 1
    nums[ptr], nums[r] = nums[r], nums[ptr]
    print("pivot: ",pivot)
    return ptr
def quicksort(l, r, nums):
    if len(nums) == 1:
        return nums
    if l < r:
        pi = partition(l, r, nums)
        quicksort(l, pi-1, nums)
        quicksort(pi+1, r, nums)
    return nums
example = []
n=int(input())
for i in range(n):
    example.append(float(input()))
print(quicksort(0, len(example)-1, example))
```

	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: 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]	pivot: 5.4 pivot: 3.6 pivot: 7.8 [2.3, 3.6, 4.2, 5.4, 7.8, 9.5]	✔

	Input	Expected	Got	
✔	4 3.2 6.4 8.7 1.5	pivot: 1.5 pivot: 3.2 pivot: 6.4 [1.5, 3.2, 6.4, 8.7]	pivot: 1.5 pivot: 3.2 pivot: 6.4 [1.5, 3.2, 6.4, 8.7]	✔

Passed all tests! ✔

Correct

Marks for this submission: 20.00/20.00.