Foundation Technical Training

Data Structures and Algorithms

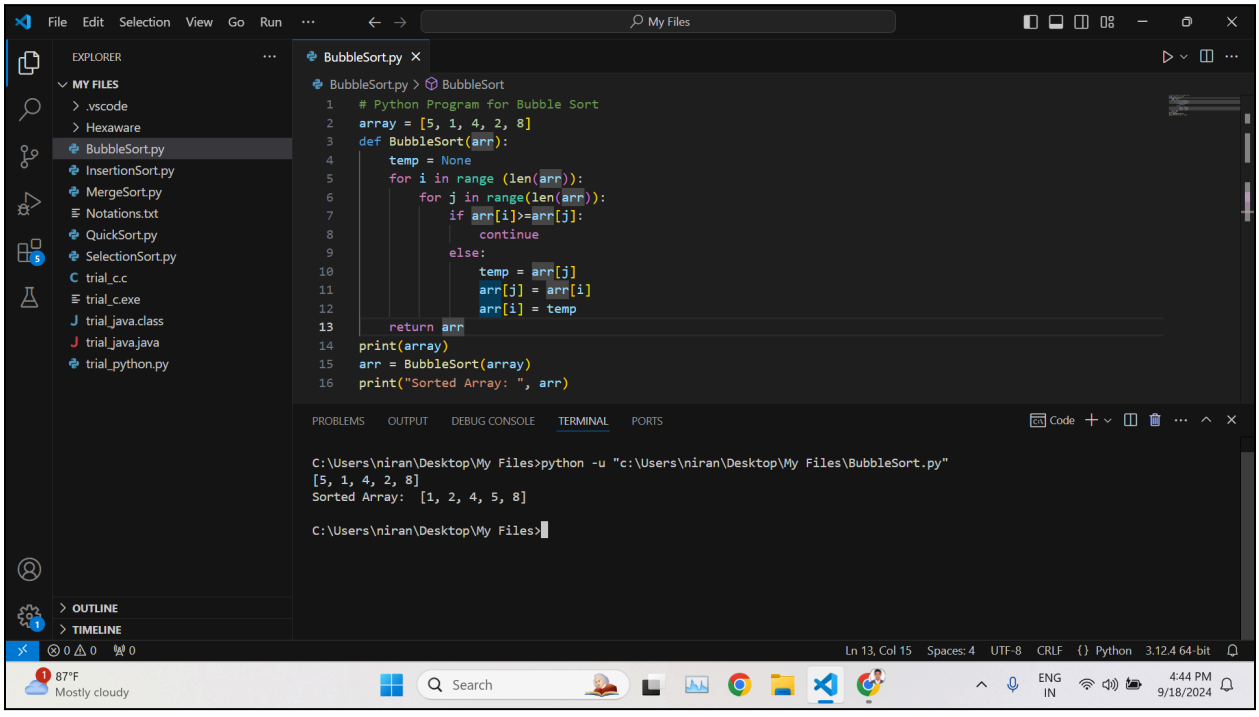
Name: Niranjan Kolpe, Batch: C#-Batch 2

1. Bubble Sort

Program:

```
# Python Program for Bubble Sort
array = [5, 1, 4, 2, 8]
def BubbleSort(arr):
    temp = None
    for i in range (len(arr)):
        for j in range(len(arr)):
            if arr[i]>=arr[j]:
                continue
            else:
                temp = arr[j]
                arr[j] = arr[i]
                arr[i] = temp
    return arr
print(array)
arr = BubbleSort(array)
print("Sorted Array: ", arr)
```

Output:

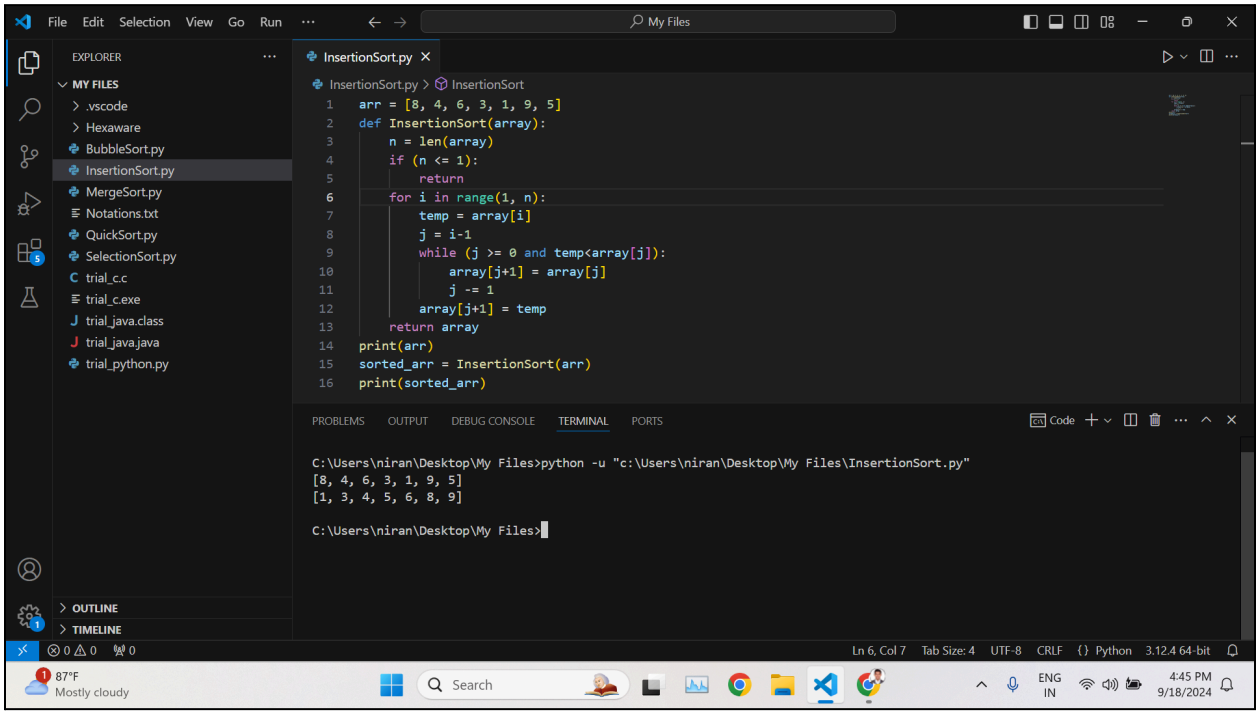


2. Insertion Sort

Program:

```
arr = [8, 4, 6, 3, 1, 9, 5]
def InsertionSort(array):
    n = len(array)
    if (n <= 1):
        return
    for i in range(1, n):
        temp = array[i]
        j = i-1
        while (j >= 0 and temp<array[j]):
            array[j+1] = array[j]
            j -= 1
        array[j+1] = temp
    return array
print(arr)
sorted_arr = InsertionSort(arr)
print(sorted_arr)
```

Output:

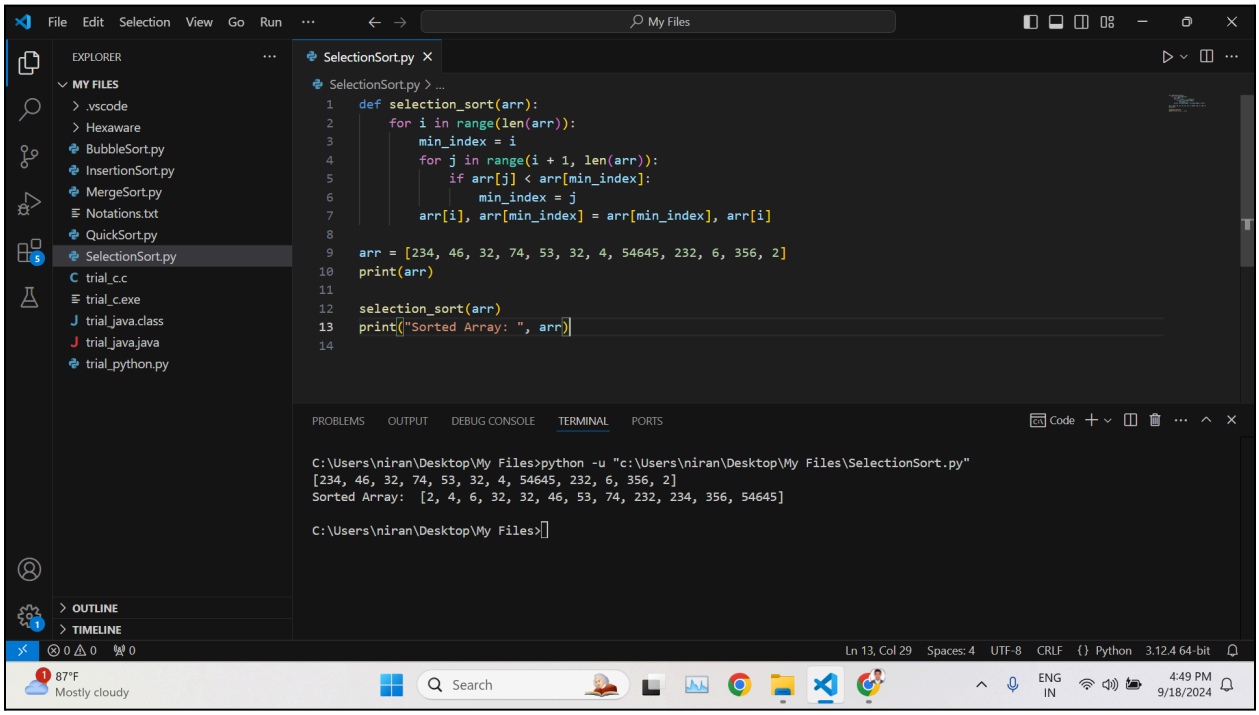


3. Selection Sort

Program:

```
def selection_sort(arr):
    for i in range(len(arr)):
        min_index = i
        for j in range(i + 1, len(arr)):
            if arr[j] < arr[min_index]:
                min_index = j
        arr[i], arr[min_index] = arr[min_index], arr[i]

arr = [234, 46, 32, 74, 53, 32, 4, 54645, 232, 6, 356, 2]
print(arr)

selection_sort(arr)
print("Sorted Array: ", arr)
```

Output:

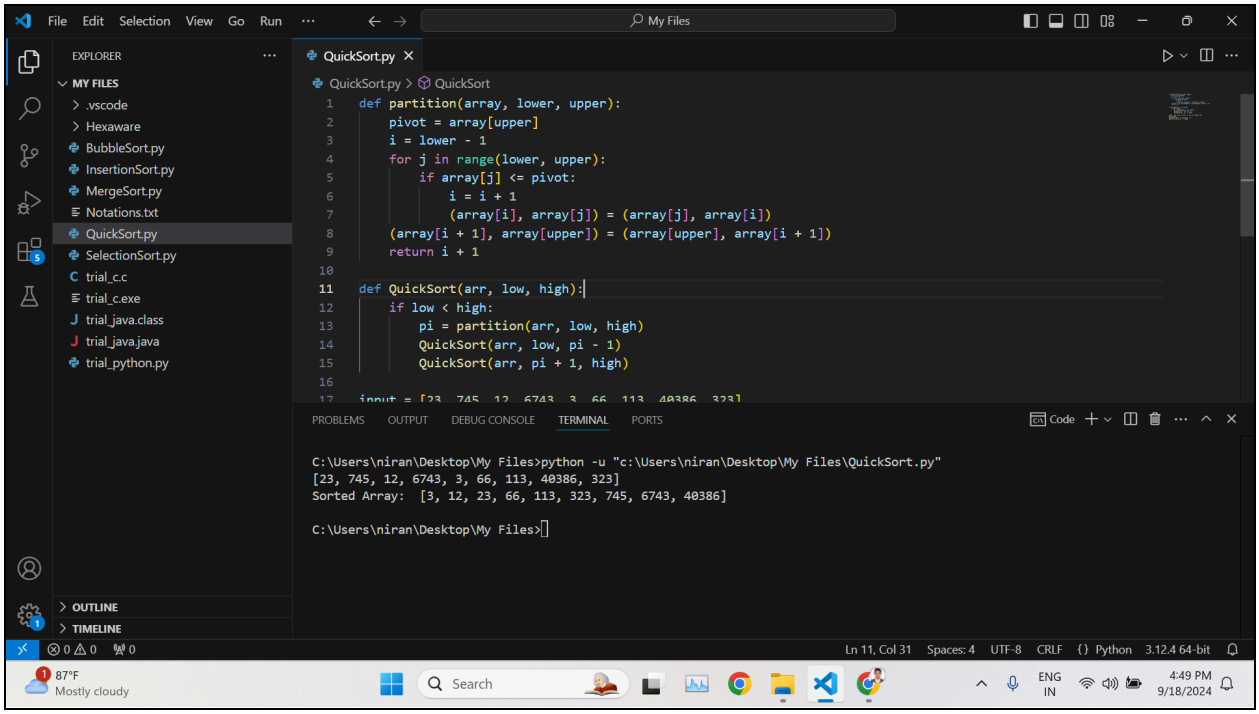


4. Quick Sort

Program:

```
def partition(array, lower, upper):
    pivot = array[upper]
    i = lower - 1
    for j in range(lower, upper):
        if array[j] <= pivot:
            i = i + 1
            (array[i], array[j]) = (array[j], array[i])
    (array[i + 1], array[upper]) = (array[upper], array[i + 1])
    return i + 1

def QuickSort(arr, low, high):
    if low < high:
        pi = partition(arr, low, high)
        QuickSort(arr, low, pi - 1)
        QuickSort(arr, pi + 1, high)

input = [23, 745, 12, 6743, 3, 66, 113, 40386, 323]
print(input)
QuickSort(input, 0, len(input) - 1)
print("Sorted Array: ", input)
```

Output:

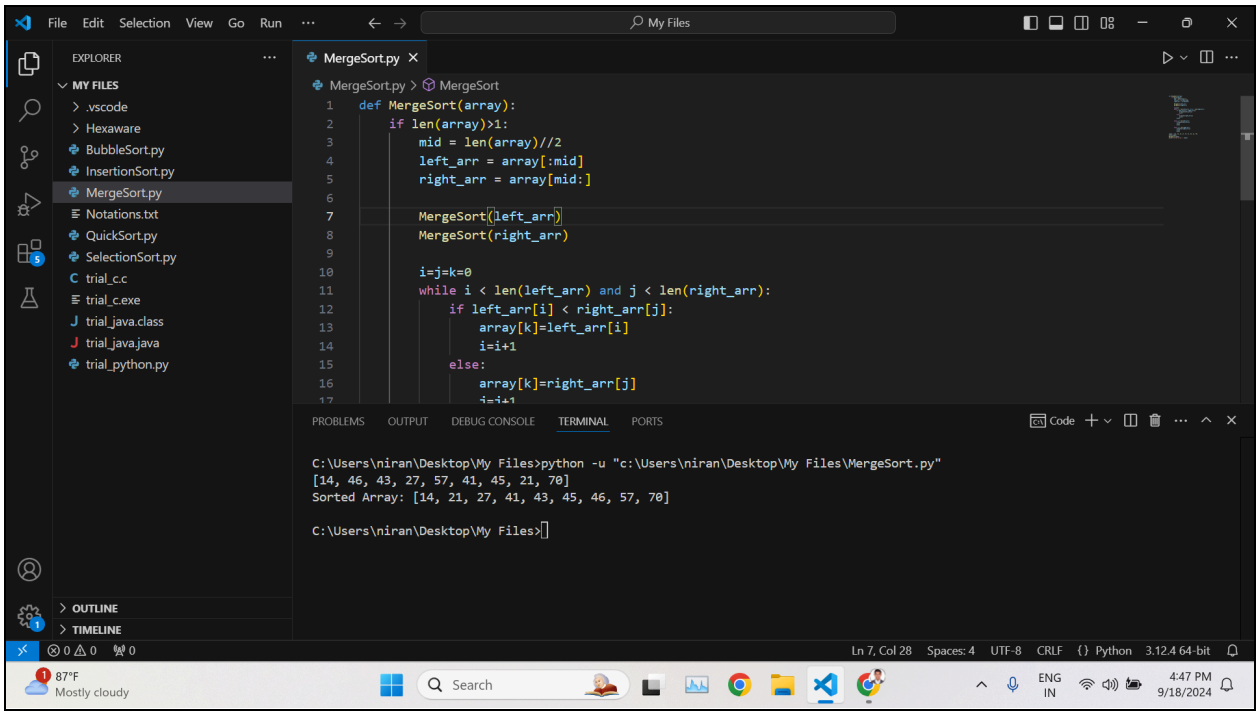


5. Merge Sort

Program:

```
def MergeSort(array):
    if len(array)>1:
        mid = len(array)//2
        left_arr = array[:mid]
        right_arr = array[mid:]

        MergeSort(left_arr)
        MergeSort(right_arr)

        i=j=k=0
        while i < len(left_arr) and j < len(right_arr):
            if left_arr[i] < right_arr[j]:
                array[k]=left_arr[i]
                i=i+1
            else:
                array[k]=right_arr[j]
                j=j+1
            k=k+1

        while i < len(left_arr):
            array[k]=left_arr[i]
            i=i+1
            k=k+1

        while j < len(right_arr):
            array[k]=right_arr[j]
            j=j+1
            k=k+1

input = [14, 46, 43, 27, 57, 41, 45, 21, 70]
print(input)
MergeSort(input)
print("Sorted Array:", input)
```

Output:

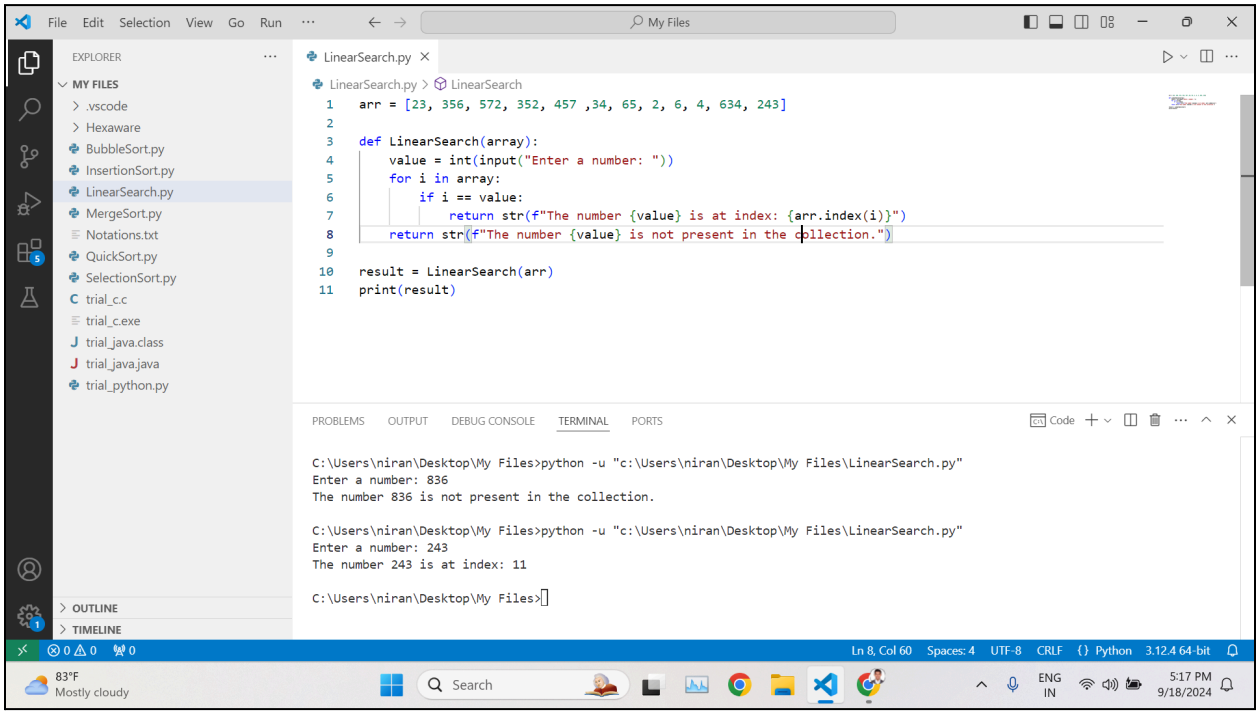

File Edit Selection View Go Run
My Files
EXPLORER
MY FILES
.vscode
Hexaware
BubbleSort.py
InsertionSort.py
MergeSort.py
Notations.txt
QuickSort.py
SelectionSort.py
trial_c.c
trial_c.exe
trial_java.class
trial_java.java
trial_python.py
OUTLINE
TIMELINE
MergeSort.py
MergeSort.py > MergeSort
def MergeSort(array):
if len(array)>1:
mid = len(array)//2
left_arr = array[:mid]
right_arr = array[mid:]
MergeSort(left_arr)
MergeSort(right_arr)
i=j=k=0
while i < len(left_arr) and j < len(right_arr):
if left_arr[i] < right_arr[j]:
array[k]=left_arr[i]
i=i+1
else:
array[k]=right_arr[j]
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Code
C:\Users\niran\Desktop\My Files>python -u "c:\Users\niran\Desktop\My Files\MergeSort.py"
[14, 46, 43, 27, 57, 41, 45, 21, 70]
Sorted Array: [14, 21, 27, 41, 43, 45, 46, 57, 70]
C:\Users\niran\Desktop\My Files>
Ln 7, Col 28 Spaces: 4 UTF-8 CRLF { } Python 3.12.4 64-bit
87°F
Mostly cloudy
Search
ENG
IN
4:47 PM
9/18/2024

6. Linear Search

Program:

```
arr = [23, 356, 572, 352, 457 ,34, 65, 2, 6, 4, 634, 243]

def LinearSearch(array):
    value = int(input("Enter a number: "))
    for i in array:
        if i == value:
            return str(f"The number {value} is at index: {arr.index(i)}")
    return str(f"The number {value} is not present in the collection.")

result = LinearSearch(arr)
print(result)
```

Output:



7. Binary Search

Program:

```
def binary_search(arr, target):
    low = 0
    high = len(arr) - 1
    while low <= high:
        mid = (low + high) // 2
        if arr[mid] == target:
            return mid
        elif arr[mid] < target:
            low = mid + 1
        else:
            high = mid - 1
    return -1
arr = [34, 745, 74, 2, 4 , 21, 44]
value = int(input("Enter a number: "))
result = binary_search(arr, value)
print(f"Element found at index {result}") if result != -1 else
print("Element not found in array")
```

Output: