Objective: Implementation of Merge Sort

Merge Sort

Merge Sort is a kind of Divide and Conquer algorithm in computer programming. It is one of the most popular sorting algorithms and a great way to develop confidence in building recursive algorithms.

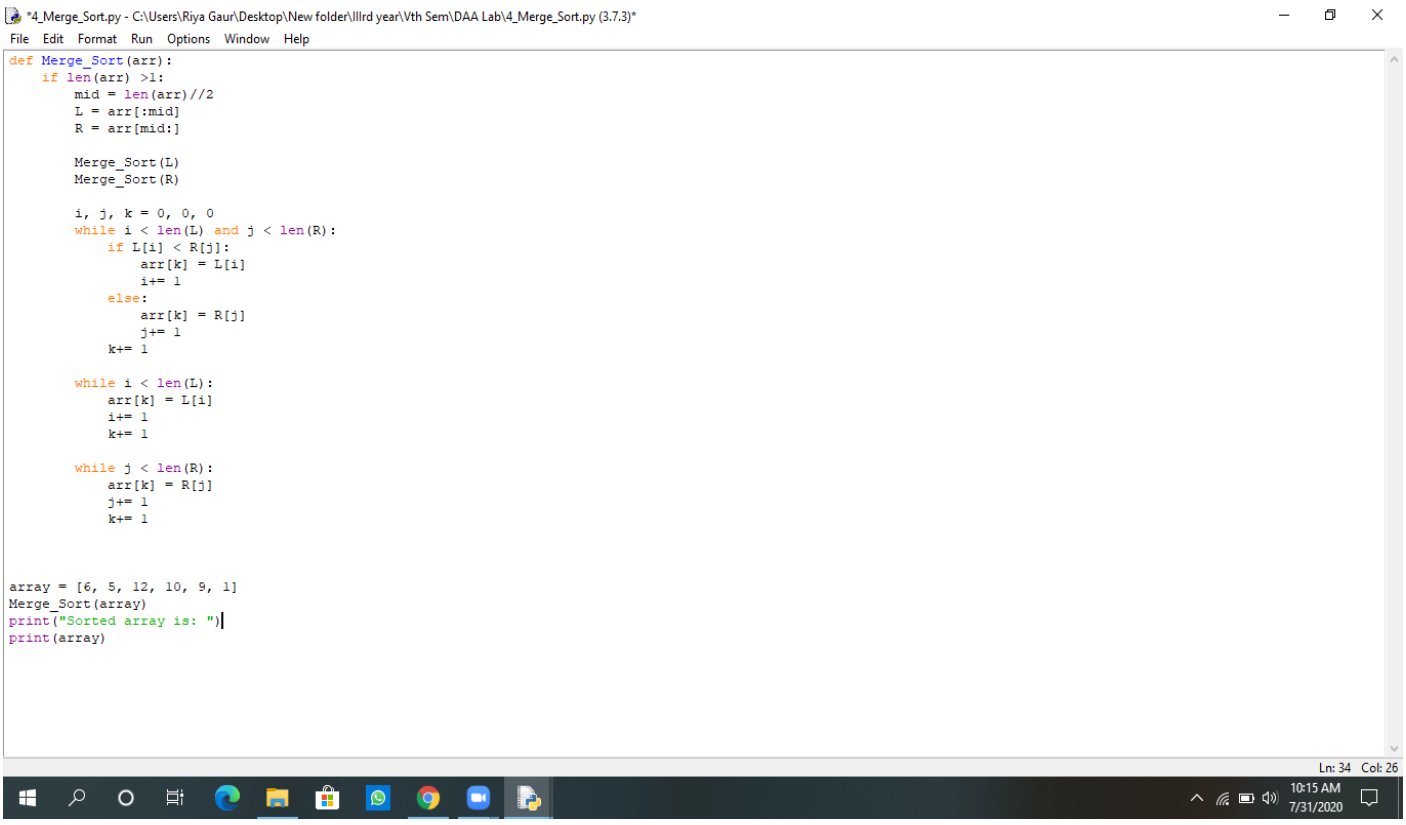
The MergeSort Algorithm:

```
MergeSort(A, p, r):
    if p > r
        return
    q = (p+r)/2
    mergeSort(A, p, q)
    mergeSort(A, q+1, r)
    merge(A, p, q, r)
```

Code:

```
def Merge_Sort(arr):
   if len(arr) >1:
      mid = len(arr)//2
      L = arr[:mid]
      R = arr[mid:]

      Merge_Sort(L)
```

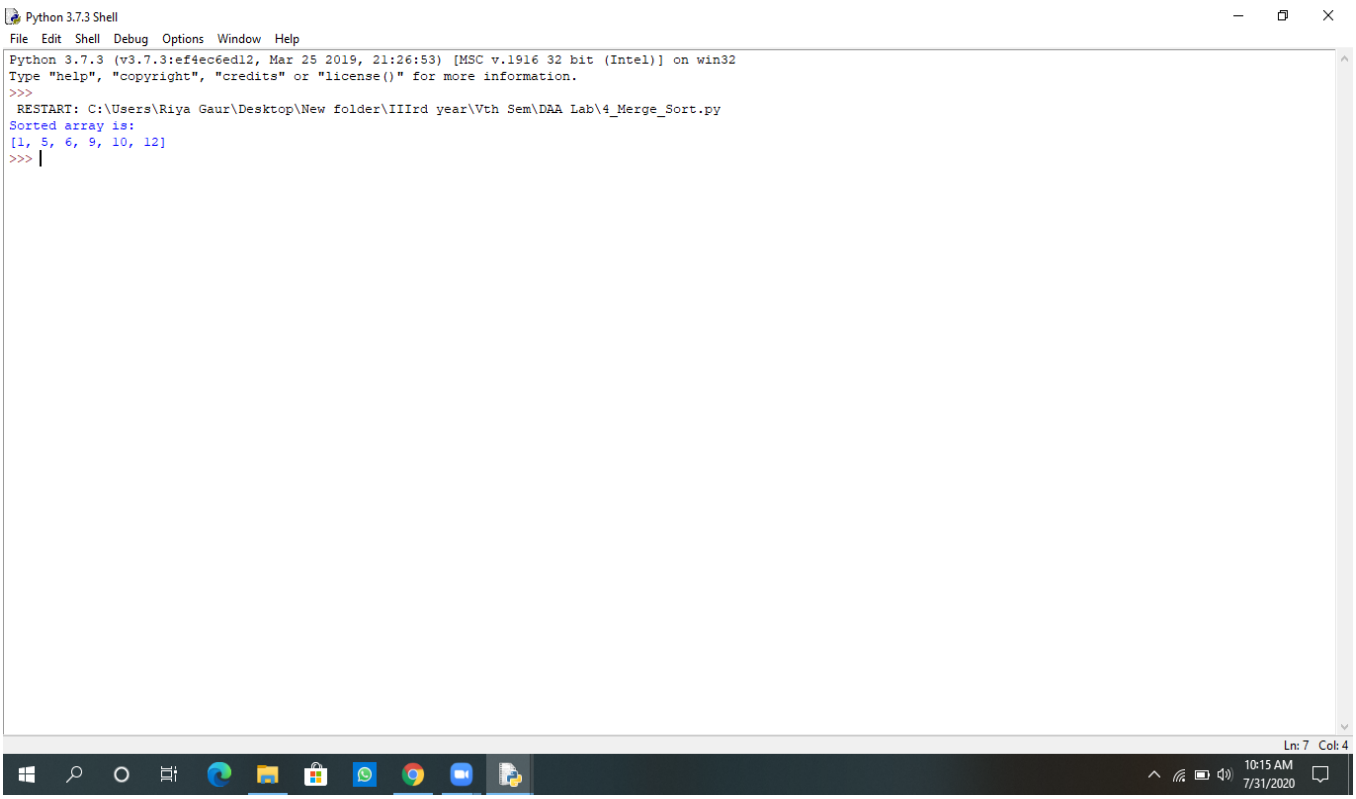
```
        Merge_Sort(R)

        i, j, k = 0, 0, 0
        while i < len(L) and j < len(R):
            if L[i] < R[j]:
                arr[k] = L[i]
                i+= 1
            else:
                arr[k] = R[j]
                j+= 1
            k+= 1

        while i < len(L):
            arr[k] = L[i]
            i+= 1
            k+= 1

        while j < len(R):
            arr[k] = R[j]
            j+= 1
            k+= 1

array = [6, 5, 12, 10, 9, 1]
Merge_Sort(array)
print("Sorted array is: ")
print(array)
```



Output:

Sorted array is:

[1, 5, 6, 9, 10, 12]



Time Complexities:

Best Case Complexity: `O(n*log n)`

Worst Case Complexity: `O(n*log n)`

Average Case Complexity: `O(n*log n)`

Merge Sort Applications

- Inversion count problem
- External sorting
- E-commerce applications