LAB REPORT

NAME : P.SATHWIKA

REGISTRATION NUMBER : 21BCE8118

SUBJECT : Design and Analysis of Algorithms.

LAB SLOT : L53+L5

Implement Merge sort using python.

1)Algorithm for Merge Sort.

Merge Sort Algorithm

Step 1: If it is only one element in the list, consider it already sorted, so return.
Step 2: Divide the list recursively into two halves until it can no more be divided.
Step 3: Merge the smaller lists into new list in sorted order.

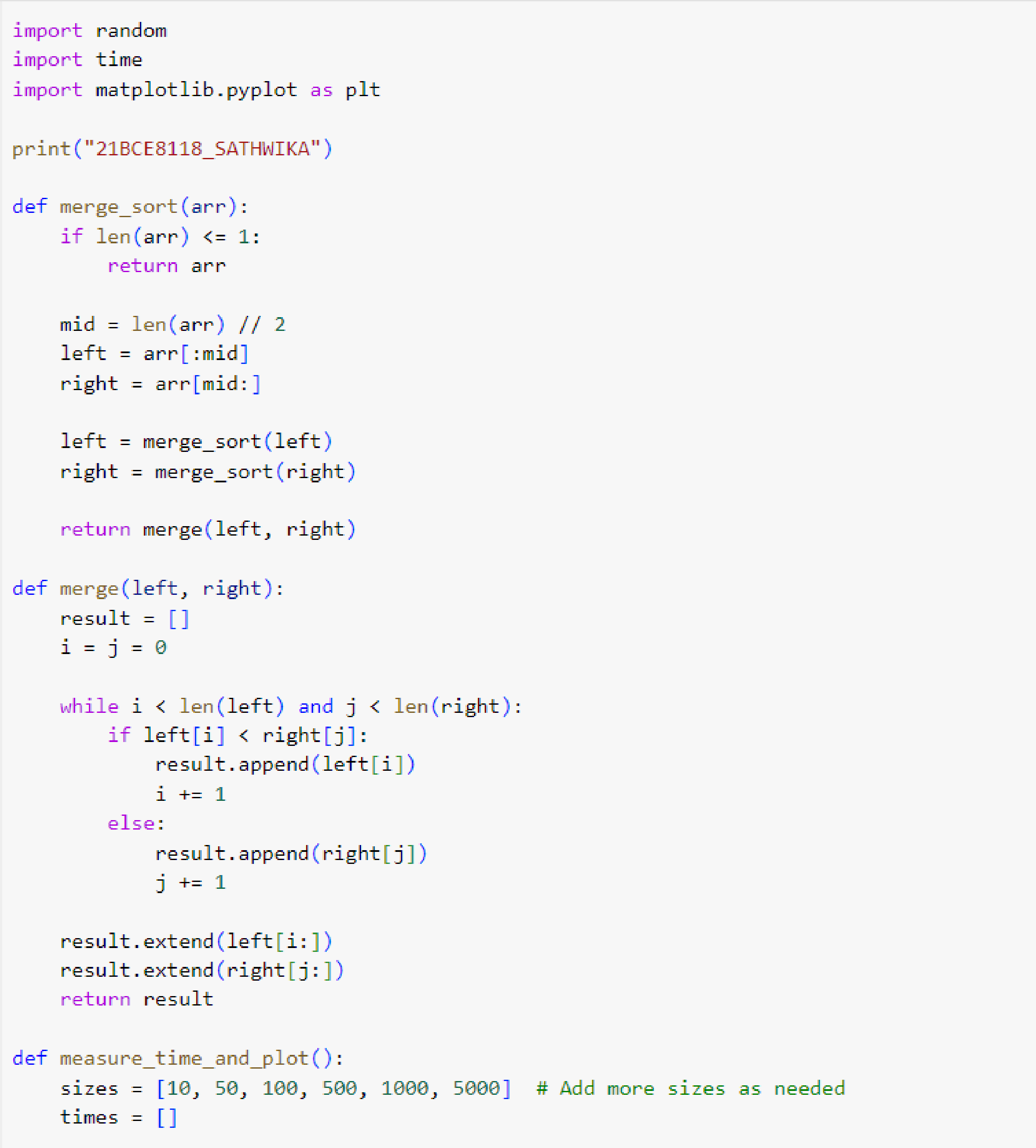
2) code screenshot along with input and output with roll no for uniqueness of student.

Code :

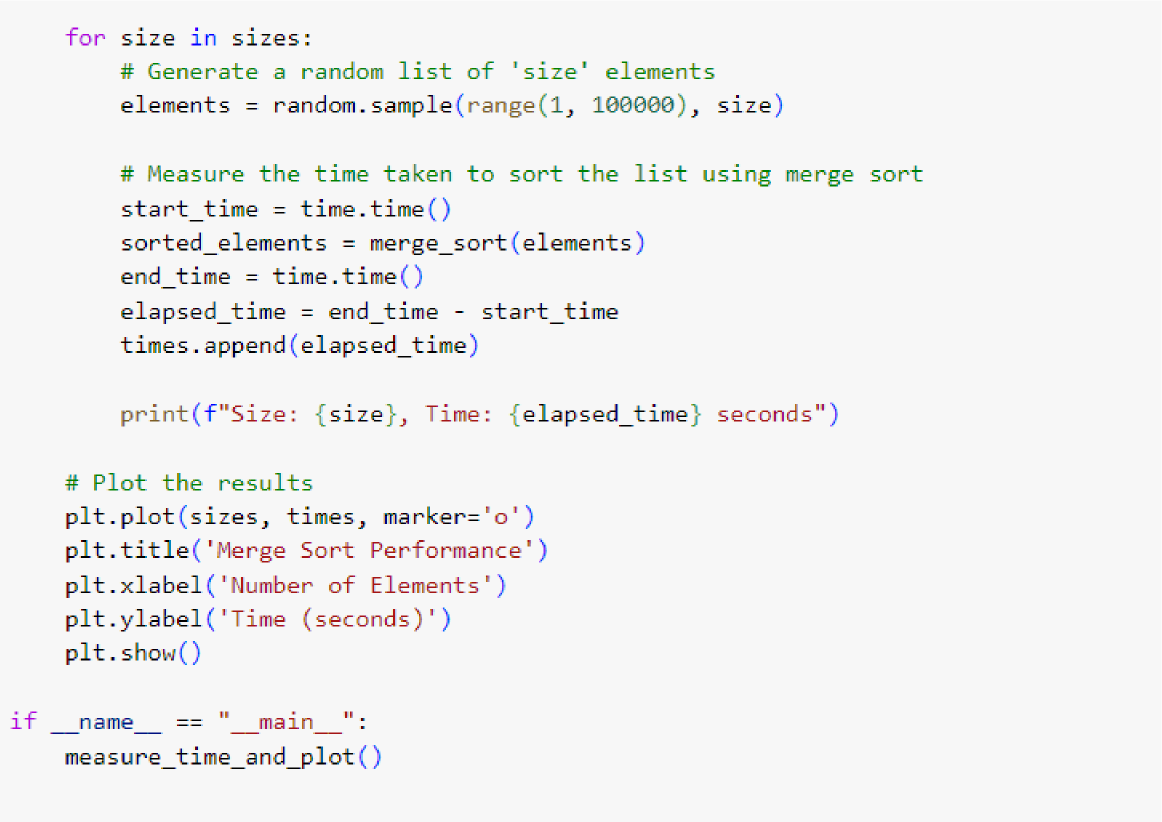
```
import random
import time
import matplotlib.pyplot as plt

print("21BCE8118_SATHWIKA")

def merge_sort(arr):
    if len(arr) <= 1:
        return arr

    mid = len(arr) // 2
    left = arr[:mid]
    right = arr[mid:]

    left = merge_sort(left)
    right = merge_sort(right)

    return merge(left, right)

def merge(left, right):
    result = []
    i = j = 0

    while i < len(left) and j < len(right):
        if left[i] < right[j]:
            result.append(left[i])
            i += 1
        else:
            result.append(right[j])
            j += 1

    result.extend(left[i:])
    result.extend(right[j:])
    return result
```

```
def measure_time_and_plot():
    sizes = [10, 50, 100, 500, 1000, 5000]  # Add more sizes as needed
    times = []

    for size in sizes:
        # Generate a random list of 'size' elements
        elements = random.sample(range(1, 100000), size)

        # Measure the time taken to sort the list using merge sort
        start_time = time.time()
        sorted_elements = merge_sort(elements)
        end_time = time.time()
        elapsed_time = end_time - start_time
        times.append(elapsed_time)

        print(f"Size: {size}, Time: {elapsed_time} seconds")

    # Plot the results
    plt.plot(sizes, times, marker='o')
    plt.title('Merge Sort Performance')
    plt.xlabel('Number of Elements')
    plt.ylabel('Time (seconds)')
    plt.show()

if __name__ == "__main__":
    measure_time_and_plot()
```

Code Screenshot :

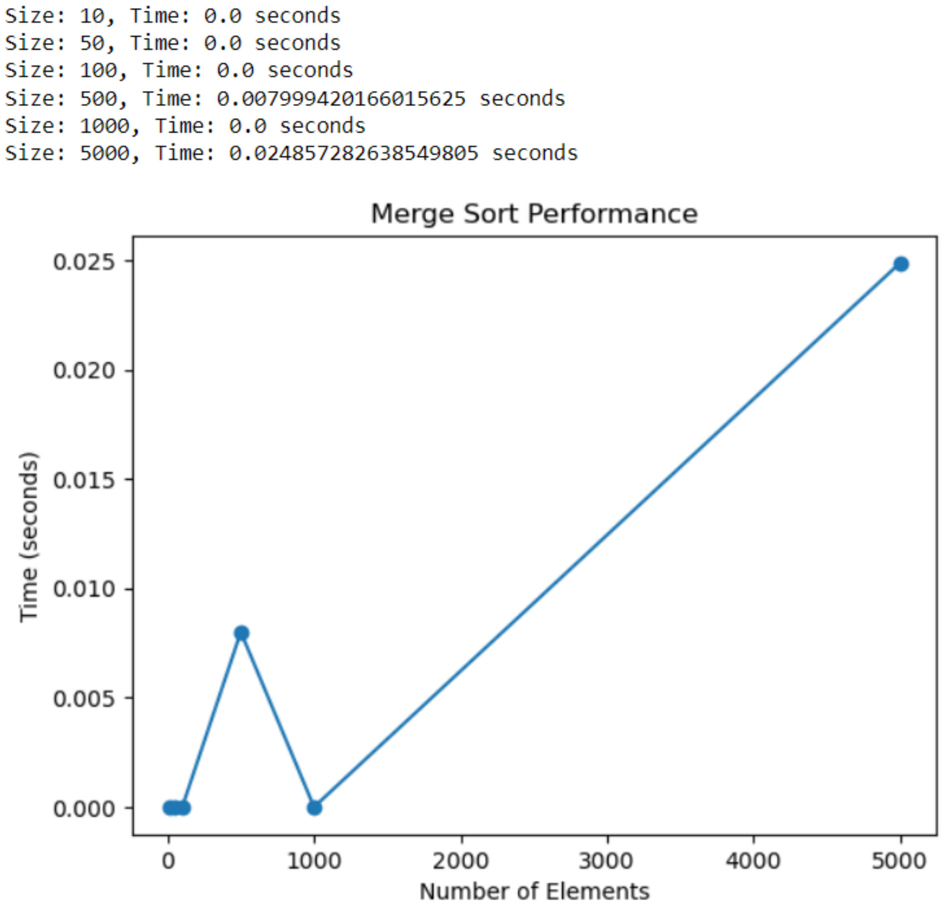


```
import random
import time
import matplotlib.pyplot as plt

print("21BCE8118_SATHWIKA")

def merge_sort(arr):
    if len(arr) <= 1:
        return arr

    mid = len(arr) // 2
    left = arr[:mid]
    right = arr[mid:]

    left = merge_sort(left)
    right = merge_sort(right)

    return merge(left, right)

def merge(left, right):
    result = []
    i = j = 0

    while i < len(left) and j < len(right):
        if left[i] < right[j]:
            result.append(left[i])
            i += 1
        else:
            result.append(right[j])
            j += 1

    result.extend(left[i:])
    result.extend(right[j:])
    return result

def measure_time_and_plot():
    sizes = [10, 50, 100, 500, 1000, 5000]  # Add more sizes as needed
    times = []
```



```python
    for size in sizes:
        # Generate a random list of 'size' elements
        elements = random.sample(range(1, 100000), size)

        # Measure the time taken to sort the list using merge sort
        start_time = time.time()
        sorted_elements = merge_sort(elements)
        end_time = time.time()
        elapsed_time = end_time - start_time
        times.append(elapsed_time)

        print(f"Size: {size}, Time: {elapsed_time} seconds")

    # Plot the results
    plt.plot(sizes, times, marker='o')
    plt.title('Merge Sort Performance')
    plt.xlabel('Number of Elements')
    plt.ylabel('Time (seconds)')
    plt.show()

if __name__ == "__main__":
    measure_time_and_plot()
```

Output Screenshot :



```
Size: 10, Time: 0.0 seconds
Size: 50, Time: 0.0 seconds
Size: 100, Time: 0.0 seconds
Size: 500, Time: 0.007999420166015625 seconds
Size: 1000, Time: 0.0 seconds
Size: 5000, Time: 0.024857282638549805 seconds
```