Elizabeth Oyebade

MET CS 526-O2

04/19/2022

**Problem 1:**

**A picture containing text, document, receipt

Description automatically generated**

**Problem 2:**

**Diagram, letter

Description automatically generated**

**Problem 3:**

**Text, letter

Description automatically generated**

**Problem 4:**

**A piece of paper with writing

Description automatically generated with medium confidence**

**Problem 5:**

**Diagram, letter

Description automatically generated**

**Problem 6:**

**Text, letter

Description automatically generated**

**Problem 7:**

Sorting Algorithms table result:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *n* | 10000 | 20000 | 30000 | 40000 | 50000 | 60000 | 70000 | 80000 | 90000 | 100000 |
| *Algorithm* |
| insertion | 23 | 159 | 139 | 253 | 428 | 494 | 608 | 819 | 1140 | 1972 |
| merge | 5 | 6 | 6 | 10 | 12 | 16 | 11 | 15 | 33 | 37 |
| quick | 4 | 2 | 4 | 4 | 6 | 19 | 7 | 10 | 14 | 15 |
| heapsort | 5 | 4 | 8 | 8 | 11 | 9 | 12 | 12 | 29 | 23 |

What I observed while running this program is that the Insertion Sort algorithm takes the longest time, and it increases the most and that is because of the way it’s implemented. The insertion sort running time is resulting in a O(n^2) because all integers to the left of the current integer have be swapped until it can be inserted into the correct place. This process is then being repeated for the next current integer. The table shows an increase in execution time for the insertion sort algorithm which is greater than the linear or O(n log n). The second diagram is showing that the quick sort is faster than merge and heap sort.