Table comparing Tree Sort with Quick Sort and Merge Sort (Note that the data and tables from quick sort and merge sort were taken from my lab 11 report):

|  |  |  |  |
| --- | --- | --- | --- |
| Array Size (in millions) | Quick Sort (s) | Merge Sort (s) | Tree Sort (s) |
| .1 | 0.0247984 | 0.0301854 | 0.0754266 |
| .2 | 0.0472827 | 0.0549248 | 0.168869 |
| .5 | 0.105937 | 0.13626 | 0.596308 |
| 1 | 0.191394 | 0.264135 | 1.66786 |
| 2 | 0.370604 | 0.546871 | 3.67577 |
| 3 | 0.584291 | 0.790418 | 6.38823 |
| 10 | 1.9604 | 2.74665 | 25.9632 |
| 20 | 4.07862 | 5.65919 | 60.7742 |
| 30 | 6.40181 | 8.43686 |  |
| 50 | 10.6707 | 14.7812 |  |
| 100 | 21.9419 | 32.1712 |  |
| 150 | 34.0513 | 51.5883 |  |
| 200 | 44.6781 | 64.4489 |  |
| 250 | 56.9538 |  |  |
| 300 | 68.605 |  |  |

Questions from lab:

1. Which tends to be the fastest of the three: Quick sort, Merge Sort, Tree Sort?
2. Which is the slowest of the three?

After another time analysis using large arrays for time sort and comparing it to the data taken from the last lab, the quick sort still appears to be the fastest of the three sorts. Unsurprisingly, the tree sort is the slowest of the three sorts. Not only does it have to go through the entire array at least once to sort it, but the tree sort in this lab also involves loading the array into a Binary Search Tree twice essentially. It calls an insert function to make the tree and then it calls a function which then calls a recursive function that calls itself as many times as necessary. This would heavily increase the time complexity it takes to run through the sort. The tree sort is still a much better candidate for large arrays compared to bubble, insertion, and selection sorts but is still slower than quick and merge sorts. However, it is important to know that tree sorts are very important since they are a way to sort arrays that are not strictly in ascending or descending order, but they are a way to sort arrays in very specific ways such as in-order, preorder, and post-order fashions which would give more variety in how data is structured in real world scenarios.