Jamie Min

Assignment 4

Randomly sorted data arrays are probably the most common occurrence as the need to sort is written in the word “random”. Overall, the sorting algorithms generally ran similar until the array size was increasing to >100k after which the simple quicksort would fall off dramatically in runtime. This test proved the notion that quicksort is generally faster in sorting than the mergesort as well with the fastest run times wavering between the Median of Three (20) and the Median of Three (100) algorithms. It is sufficed to say that either the base case 100 or base case 20 would be the most applicable to any real-world situation involving randomly sorted primitive data types.

As for the already sorted array, the tests proved that the mergesort would run in its best-case scenario beating out the quicksort. The worst-case scenario for quicksort is demonstrated here. Lastly, the reverse sorted array shows the opposite. The Median of Three (100) proved to be the most efficient algorithm while the mergesort was the worst.

Outside the scope of these two data arrays, the simple quick sort had the worst run time but were not included in the analysis due to the significant average time difference in the results.

In conclusion, when dealing with randomly sorted types and data stability is not a concern, Median of Three (100) or Median of Three (20) proves to be the most efficient in sorting data.