**With Regards to Big O**

Our results were not too surprising, and aligned well along Big O predictions. If we were to use our average time of 16000 on the quickSort algorithm as the basis for predicting the performance of the bubbleSort and insertionSort algorithms, we would expect somewhere around 13 million clock ticks. However, it more than likely the case that quickSort and mergeSort outperformed their own Big O worst case scenario. Therefore, we can see that values in the millions of clock ticks are not unexpected for this type of scenario. The stark difference between n^2 and n *log* n algorithms is very drastic and emphasizes why we should take care when choosing an algorithm for sorting purposes.

**Inconsistencies and Surprises**

If you take look at our average times, you will see that mergeSort outperformed quickSort. This is more than likely due to the fact that our version of quicksort used the first index as the pivot point, which is not the most efficient and very likely degraded performance. Performance was also likely further degraded because of the lack of unique random values and the smaller range in which the random values had to lie. It is likely that if changes were made to address these issues that quicksort would marginally outperform mergeSort.