The time difference between all the sorting methods were more drastically different than I originally expected. I thought they would relatively close in terms of runtime but after sorting the text file with 100,000 elements, it really showed the time differences between the three sorting methods I chose. The merge sort method was significantly better than selection sort and bubble sort methods, and the bubble sort method is significantly worse than the merge and selection sort. There are a lot of tradeoffs between picking one algorithm over another. Bubble sort is easy to implement, but does not do very well with lists with a high number of elements. Selection sorts do really well when sorting through short lists and does not take up extra memory space, but also does not do well with a large number of elements and is influenced by the initial ordering of items before the sorting process. Merge sorting is the fastest out of all of these methods, but requires more memory space. Depending on what programming is used both the time complexity and space complexity of a sorting method can be different from one language to another. I think the shortcomings of this empirical analysis is my lack of experience and current knowledge of sorting algorithms. The things I noted are just from my basic analysis from what I have learned in class and how my program worked out, so they may not be entirely accurate. The more I work with sorting algorithms and learn either online or in another class the more my future empirical analysis will be more accurate and better.