Michael Masakayan
CPSC 350-3

For this project I expected the times to very alot but in my cod it seemed like the times were very very close and sometimes I couldn't see the difference because of the decimal places in the time. I think that because we are using a optimised language and using code that has been optimised throughout the years that it is hard to notice the differences given the small data set. It seemed that even if I made the test file to be over a couple of thousand of numbers it chugged through the numbers and sorted them super fast. From general understanding I thought they would very greatly but after looking into the time complexities I realized that they actually in fact do not vary as much as I though. They all share the same worst case in terms of time complexities and the only real difference is their best case scenario for insertion sort,selection stort, and bubble sort. For merge sort I knew that the worst case scenario is O(nlogn) and quick sort was O(n2). In terms of merge sort versus quick sort quick sort is more likely better one arrays that are smaller while merge sort is ok with any array. Quick sort is less efficient than merge sort is . For insertion sort when the levels are larger it starts become less and less efficient. For selection sort the time complexity mostlyu stays the same and it dosn't have a better time complexity. And bubble sort is a less efficient slower algorithm. For Empiracal analysis the shortcomings include not knowing exactly what you are working with. In theory a certain algorithm can work better and run at its best case scenario given a certain type and amount of data. But this will very based on external factors like if you can use recursion or if you have a memory limit. So it is a great tool to see what algorithm would be best for your data in theory but in practice you need to know what will be best for a given situation. It is kind of an estimate for what your code will do to the data set it will run so it should be taken as an estimate and not an end all be all. All in all I expected their to be alot more of a difference when switching the

algorithms because in the book it makes it seem like its a huge difference. But it is easy to forget that when you learn about these algorithms that it is talking about best case, worst case, and average times of the algorithms. It all depends on what you are sorting through, how much you need to sort through, and external factors that might hinder what you are sorting throug.