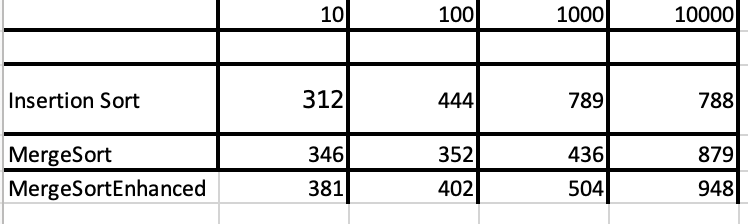
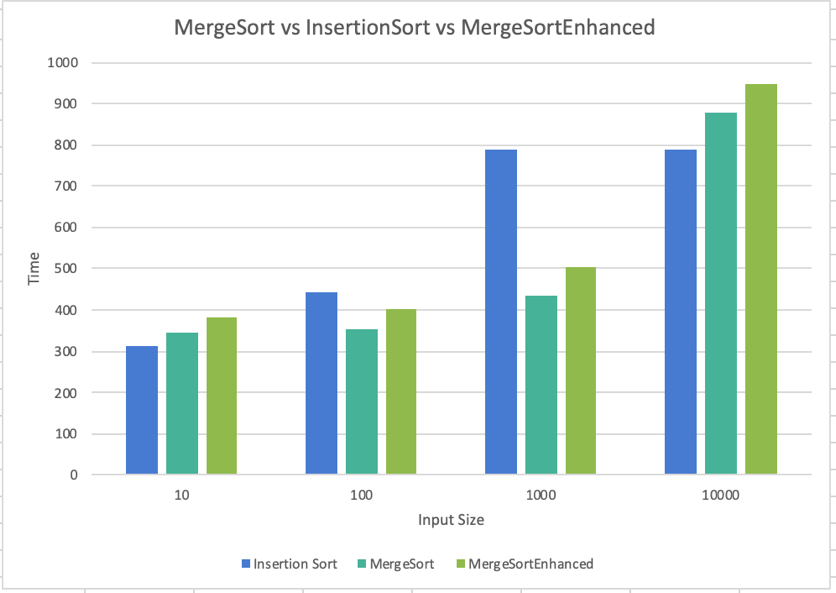
Practical 5 :

**Quick Questions:**

1. Linearithmic time
2. It uses extra space in proportion to the size of the input
3. Divide and conquer
4. Insertion Sort
5. Merge sort

**Exercise 3**:





As we can see from the results, insertion sort is always faster than merge sort and merge sort enhanced for all input sizes. Insertion sort has a fast best-case running time and is a of good sorting algorithms to use. This is evident in the graph. For the testing I used a random, unsorted array, but for when the input list is mostly sorted, insertion sort can work even better. Since the array sizes weren’t extremely big either, insertion sort works better as it has far fewer operations per single exchange. The results for mergeSortEnhanced were not far off mergeSort, as the times were pretty close as we can see from the data. mergeSortEnhanced used insertionSort within its code to make it more effective.