GIT Department of Computer Engineering
CSE 222/505 - Spring 2020
Homework 6 Question 2 Report

Melihcan Çilek 1801042092

1. CLASS DIAGRAMS

Class diagrams added as addition named 1801042092_Q2_Class_Diagram.png because class diagram of question 2 was so wide, it could not be fit to pdf file.

2. PROBLEM SOLUTION APPROACH

In this problem, we asked to write sorting algorithms at the book and redesign two of these algorithms which are Merge Sort and Quick Sort as they both will be implemented as LinkedList<T> generic type for data type. Implementing Merge Sort is not that much hard because when we make partition for arrays, we can go on linkedlist that we defined at the beginning by using iterators. Iterators was really helpful while I'm going this homework. Unless I use iterators, I would get much worse results about running time. Reason that I prefer iterators is that we had to use get() method of linkedlist alot when implementing this part so running time of the get() method at linkedlist is O(n) so it takes so much time but when implementing this by using iterators, running time of iterators is O(1) which is constant time so I implemented and tested really easy for merge sort. If we come to Quick Sort side, First, I did the implementation by using get() methods and because we are doing so much swap in quick sort, everytime program wants to swap two of the indexes, it was iterating till that indexes and swap so it was taking so much time. After I figure this thing out, I changed my impelementation from get() methods to iterators.

3. TEST CASES

Case	Scenerio	Test Steps	Expected
Test Sorted Data	Sorting Sorted Data	Giving Algoritm sorting data	Input will be sorted according to algorithm given
Test Random Data	Sorting Random Data	Giving Algorithm Random Data	Input will be sorted accoring to algorithm given

4. RUNNING AND RESULTS

If you run this program, you will get output with an .csv file that contains milliseconds that takes for sorting these algorithms. There are graphs named 1801042092_output.csv that compares worst and best case running times and that I obtained from this implementations.