Name: Tay Ho
Cpts 233: PA #1 – Benchmarking a LinkedList
Date 09/18/20
The Report

A: Problem Statement.

The Goal of the problem is to insert all of the number from the txt file into a linked list in a sorted way.

B. Algorithm design.

There is a no difficult algorithm design, but the challenging part is the implementing of the code. Like how it is best read the file and using the linked list method. On the other hand, there are only a few ways to compute the out values such as 'min', 'max', and 'med'.

For example, to find the minimum number in the whole list, I start with the first number on the list, compare it to the rest of the number on the list one by one, and if I find any number that is lower than the one that is in hold for I change it to a new minimum number.

Method to find maximum number is the same with minimum number. I start at the first number, go through the list and look for any number that is highest in the list.

Finding median is much fast, if the list is odd then I divide the list size by two and get number the location of middle of the list. And, if the list is even than I find the number of the left middle of the list and number at the right middle of the list. Then, added them together and divided by 2. Thus, we find the median of the linked list.

These methods of finding 'min', 'max', 'median' definite can be improved to optimize run time. Since the linked list is sorted, to find 'min' and 'max' I can apply "divide and conquer" method to lower the time complexity of the function from O(n) to O(nLog(n)). This method is way faster and much more efficient.

C: Experimental setup:

- Machine specification –
  - CPU – Intel Core-i7-6700k @ 4GHz
  - RAM 8.0GB
  - Samsung SSD 1TB
- The code have the same run time on all three experimental run.
- I used Ubuntu(64-bit) OS inside a virtual machine and I compile the code in Visual Code Studio (version: 1.48.2)

D: Experimental Results & Discussion

While writing the code, I write a text file with ten to twenty number in it to help simplify the data and easy on the eye. However, after I run the bigger data input1.txt file, there is a noticeable slow time than before; it was no longer produce an instant result. For example, the time it took to insert all the number is 524 milliseconds. In addition, compare the input2.txt (786.4 kB) to input1.txt(5.7kB) more than 157 times in size. It took my computer forever to run.