CS2302 Data Structures

Fall 2019

Lab Report #3

Due: October 4th, 2019

Professor: Olac Fuentes

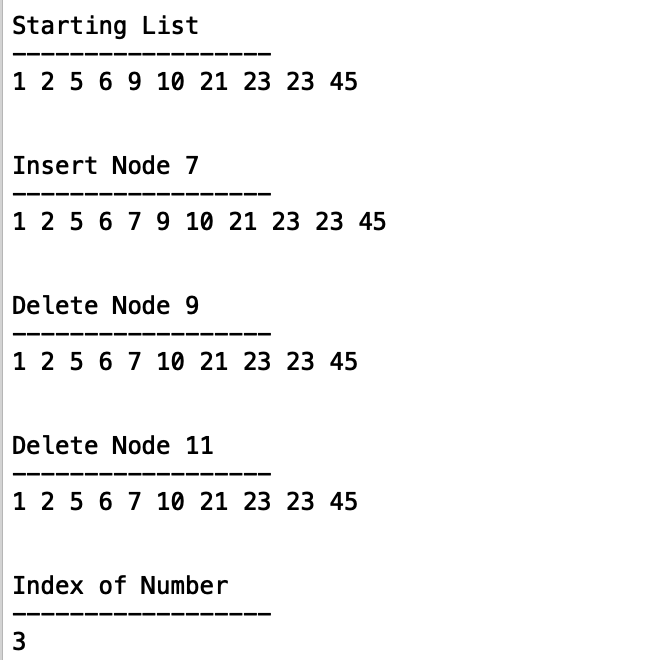
TA: Anindita Nath

**Introduction**

The problem presented was to implement a class called sorted list that acted just as a regular class list except it was always sorted. Then within that class we have to implement the functions for Print, Insert, Delete, Merge, IndexOf, Clear, Min, Max, HasDuplicates, and Select.

**Proposed Solution Design and Implementation**

Design –



This is the output that the user will see when running the program, further down in the output code from a regular list is ran from methods that differ in code from a sorted list class so that the time complexity can be done when in comparison of the two classes.

Operation #1: I implemented the class Sorted List that gets sent sorted lists only so that the original list added is always sorted, then methods such as insert were tuned so that they insert nodes that keep the list sorted. Similarly, the function Merge had to be implemented so that the two sorted lists merge into 1 sorted list which is different from the regular list which would just add the new list to the end. Some methods such as Min, Max, and HasDuplicates were much easier to implement since the list was always sorted.

Operation #2:

|  |  |  |
| --- | --- | --- |
| Function | Sorted List | List |
| Print() | O(n) | O(n) |
| Insert(i) | O(n) | O(1) |
| Delete(i) | O(n) | O(n) |
| Merge(M) | O(n^2) | O(n) |
| IndexOf(i) | O(n) | O(n) |
| Clear(i) | O(n) | O(n) |
| Min() | O(1) | O(n) |
| Max() | O(1) | O(n) |
| HasDuplicates() | O(n) | O(n^2) |
| Select(k) | O(n) | O(n) |

**Experimental Results**