Tony Diep
2/23/17
CS 2420
Algorithms and Data Structures

Design Document: The Linked List

The purpose of this project is to introduce and implement the Linked List data structure and how they relate to the ArrayList data structure. We are familiar with the ArrayList since we first learned about it in CS 1410. The LinkedList is the first "linked" structure we will learn this week as we will eventually explore other linked structures throughout the semester. It essentially comprises of a chain of "nodes," which are boxes that contain an generic element and a reference to another node. One can visualize and think that these linked structures can be implemented using recursion. Additionally, we can mentally note that the linked list data structure is also related to the ArrayList implementation. One reason is because they are both type of Lists which implement the List interface. All classes that implement this List interface must have various methods to be required. However, our List interface will be somewhat distinct from Java's built-in List interface. For example, we will have methods such as but not limited to "addFirst()," "addEnd()," "reverse()" and a few other recursive methods such as but not limited to "compute_size_recursive()" and "toArrayList_post_recursive()." We can see that recursion will be a useful mechanism for this assignment. Thus, this project emphasizes linked lists and recursion because they complement each other well.

There are several classes to be created and implemented for this project. First, we have the List_2420 interface which represents our own implementation of Java's built in List interface. Next, we have the Array_List_2420 class, which represents our own implementation of Java's built in ArrayList class. We can recall that the ArrayList is essentially a "dynamic array" which we have learned in CS 1410. Of course, the assignment specifications asks us to not utilize the built in class except for implementing the toArrayList() method. It will be implementing the List_2420 interface which represents our own implementation of Java's built in List interface. Then we have the Linked_List_2420 class, which represents our own implementation of Java's built in Linked List interface since we are not allowed to use Java's built in classes. This Linked_List_2420 class will also have a helper static class called "Node" which represents a single node that constructs a linked list. Finally, we have the List_2420 Test class which will provide general testing for the List_2420 interface as well as provide testing for the Array_List_2420 and Linked_List_2420 classes.

When approaching this assignment, we are required to do the "Test First" rule, which means we must implement JUnit Tests first before implementing any of the other classes. This way, we are able to provide a good idea of where our methods should go in terms of correctness. However, we must also provide a Timing class that performs timing experiments after we have completed and corrected the implementations for all of the classes. Additionally, we are also required to provide pictures of what the linked list data structure looks like when it is adding or removing links and data in and out. Although it will take immense work, we will be able to understand how linked lists and other data structures conceptually work because having visual representations will certainly benefit us and thus we can easily code them and take less time than necessary.