Name – Rodas T. Gebreslassie

Date – 5/23/2018

Class – CSC 143

**SortedIntListTest.java**

SortedIntList 1 :

S1: [1,3,5,5,7,7,8,9,10,11,12,15]

S1 after removing element at index 3: [1,3,5,7,7,8,9,10,11,12,15]

Element at index 5: 8

The size of the SortedIntList is: 11

s1.isEmpty() = false

SortedIntList 2:

s2: [1,3,4,4,5,5,7,7,7,10,11,12,14,14]

s2.size (): 14

s2 after setting unique to true: [1,3,4,5,7,10,11,12,14]

s2.getUnique () = true

s2 after adding 3 with unique set to true: [1,3,4,5,7,10,11,12,14]

The largest element in the SortedIntlist s2 is: 14

The smallest element in the SortedIntList s2 is:1

SortedIntList 3:

s3: [1,2,2,2,2,3,3,5,5,5,6,8,8,10,11]

s3.indexOf(6) = 10

s3.count(2) = 4

s3.indexOf(4) = -8

**Reflective Journal**

The first thing I did when I start this project after the constructor methods is writing the add method. First I write the method in a way that it doesn’t track where a certain value should be added, it is somewhat similar to the add method of the ArrayIntList class. Then I modified it in a way that it can track and spot the position where a single element should be added by writing a private method compare. The private method compare() will accept a value and return the position in which a single element should be added in a list. By calling the method compare() from the method add() we can find the appropriate position for a single element to be added in the list.

Next I have to write toString() method to check if the method add() work properly. Once I make sure the add method work properly, I move to the indexOf() method, I used binary search to find the position a specific value is found. Once the binary search find the index of a value then I used while loop to find the position of the first occurrence of a given value.

The method took me a lot of time to get it to work count() method. By the time I was writing the method count I didn’t know that the method indexOf() should return the position of the first occurrence of a given element in a list. So I have to check the elements to the left and right of the index(returned from the method indexOf()). To count a given value that is found at the middle of the loop this method work find but when I try to count value that is at the front of the list or at the end I keep getting index out of bound errors. So I have to make many adjustments to resolve this error.

Once I make sure that the sortedIntList class works well with the list that accept duplicates then I modify the some methods in the class so that it can work when the set unique switch set to true.