Project 2

CS2400-002

By: Alexander Sanna

Due: October 7th, 2022

Section 1: Project Specification

The SetInterface<T> interface:

SetInterface<T> is a generic interface that serves to stand as a virtual set of integers. The SetInterface allows a programmer to implement a Set ADT by implementing this interface. The definitive aspect of the Set is that it doesnt allow for repeat data to be entered in. For example: If you try adding “5” to a set A: {2, 5, 10}, the result will be {2, 5, 10}. The set will not allow for multiple entries of the same value. Another key aspect in regards to the SetInterface is that data is not organized in any particular order. Similarly to the ArrayBag, order of the data is not important as the methods in Set ADT will account for locating specified entries in the set.

Interface SetInterface<T>

```
public interface SetInterface<T>
```

Method Summary

All Methods | Instance Methods | Abstract Methods

Modifier and Type	Method	Description
boolean	`add(T newEntry)`	takes data as a param and adds to the set.
void	`clear()`	clears the entire set.
boolean	`contains(T anEntry)`	scans the entire set to see if the parameter is present
boolean	`equals(SetInterface<T> setB)`	scans 2 sets and returns boolean indicating whether the set calling this method is equal to the parameter
int	`getCurrentSize()`	returns the size of the SetInterface
boolean	`isEmpty()`	returns whether the setInterface is empty
T	`remove()`	removes the first entry of the set
boolean	`remove(T anEntry)`	takes in data as a parameter and removes s it from the set.
boolean	`subset(SetInterface<T> setB)`	scans 2 sets and returns boolean indicating whether the set calling this method is a subset of the parameter
T[]	`toArray()`	converts the set into an array and returns the array
String	`toString()`	returns the string of whatever state the SetInterface is in
SetInterface<T>	`union(SetInterface<T> setB)`	takes a setInterface as a paramter and adds it to the beginning of the set calling the method.

LinkedSet:

The LinkedSet class is the implementation of the SetInterface interface. The LinkedSet serves to stand as a Set of type T, where T is specified by the client. The LinkedSet implements every method in the SetInterface interface as seen above in the JavaDoc. Along with those methods, the LinkedSet has an inner class Node that serves as the object utilizing the LinkedList ideology to store data. Each Node

contains a reference to its own individual data, along with a reference to the next Node in the list of data within the LinkedSet. The LinkedSet class also has a constructor to create the LinkedSet to begin with:

LinkedSet()

Takes no parameters, sets the FirstNode reference to null and the initial number of items in the set to 0.

Node:

The node class is an inner class of LinkedSet. It serves to store the references to the data along with reference to the next Node within the LinkedSet.

Section 2: Testing Methodology

To test the methods in my LinkedBag ADT, I used the SetTest.java. Every single method that is seen in the SetInterface was tested with multiple instances including different set lengths, different set values and in some instances empty sets.

Examples of these tests written in the method include but arent limited to:

```
System.out.println("Set A Not Empty: " + !set.isEmpty());
```

Should return true.

```
  System.out.println("Set A current size: " + set.getCurrentSize());
```

Outputs the current size.

```
System.out.println("Add 5 to set A: " + set.add(5));
System.out.println("Set A now contains: " + set.toString());
System.out.println("Remove 5 from set 1: " + set.remove(5));
System.out.println("Set A now contains: " + set.toString());
```

Adds 5 to the set, shows its been added, removes, shows its been removed successfully.

```
System.out.println("Set 2 contains 5: " + set2.contains(5));
```

Reads false, since 5 has been removed

```
set2.clear();
        System.out.println("After clearing both sets, Set A: " + set.toString());
        System.out.println("empty returns: "+ set.isEmpty());
```

Clears the set and shows that it has been cleared.

```
System.out.println("Set A subset Set B: " + set.subset(set2));
        System.out.println("Set A equals Set B: " + set.equals(set2));
        System.out.println("Set A union Set B: " + set.union(set2).toString());

        SetInterface set3 = set.union(set2);

        Object[] setOfBoth = set3.toArray();
```

Tests all subset, equals and union methods and shows they work by using the toArray().toString().

Section 3: Lessons learned

Through this project I became a lot more familiar with the LinkedList data types, as well as basic java programming that I can confidently say took me by surprise. This program was the first time I had ever used .concat() in my entire computer science history. The StringBuilder class as well. I also utilized the theory of inner classes with the Node class implementation. This was a trying project, as I spent easily over 10 hours writing it all, but I'm extremely happy with the results and I feel that much more confident as a programmer overall.