**Homework #3**

**Generics & Linear Structures**

total points:

0All of the code used in this assignment is available in the hw03 package.

Question 1. 25 25( points) Create a generic class named Pair in your hw03 package. The Pair class is able to store and retrieve two values. The types of these values must be specified by type parameters. The Pair class must include the following methods:

* A constructor that allows the two values to be specified as parameters.
* getFirst – returns the first value
* getSecond – returns the second value
* setFirst – changes the value of the first value
* setSecond – changes the value of the second value

It is not necessary to write any comments or tests for the Pair class.

Question 2. 15 40( points) Add a main method to your Pair class. In that method define the following variables and objects:

i. A variable intPair that refers to a Pair object that holds two Integer objects.

ii. A variable mixPair that refers to a Pair object that holds a Double object and a String object.

iii. A variable pairPair that refers to a Pair object that holds two Pair objects, one as defined in part i, and the other as defined in part ii.

Question 3. 10 50( points) Copy the CS132Queue interface from the linear.object package in the COMP232 sample code to your hw03 package and rename it to CS232Queue. This interface uses Object as the type for the elements on the stack. Modify your new CS232Queue interface so that the type of the elements that the queue can hold is specified using a type parameter. Note: You do not have to implement a queue, just rewrite the interface so that it is generic.

Question 4. 25 75( points) Create a CS232ArrayStack class in your hw03 package that implements the CS232Stack interface and uses a CS232ArrayList as the backing store. Use the methods of the CS232ArrayList to implement the stack operations. Try to make the stack operations as efficient as possible.

Question 5. Complete the implementation of the following methods that appear in the CS232DoublyLinkedList class in your hw03 package:

(a) 10 85( points) remove()

(b) 10 95( points) clearTo()

(c) 15 110( points) addAllAt()

Question 6. Complete the implementation of the following methods that appear in the DLLIterator inner class in the CS232IterableDoublyLinkedList class in your hw03 package. Make sure to consult the javadoc in the CS232Iterable interface carefully, since that is the only way you can know how these methods should behave correctly.

(a) 10 120( points) hasPrevious()

(b) 10 130( points) previous()

(c) 15 145( points) remove() – Hint: use a field in the iterator to keep track of the node returned by the most recent call to next or previous and use that to determine if the call to remove is valid, and if so what element to return.

Question 7. (Ungraded) Modify the COMP232ArrayList class in your hw03 package so that it implements the parts of the CS232Iterable interface as described below. You will need to:

i. Make COMP232ArrayList implement the CS232Iterable interface.

ii. Add an inner class to CS232ArrayList that implements the CS232Iterator interface. This class should support the hasNext, next, hasPrevious and previous methods of CS232Iterator interface. You do not need to implement the insert or remove methods. Have these methods throw an UnsupportedOperationException. Hints: The cursor can simply be an integer. Take advantage of the get method in the CS232ArrayList class when implementing the iterator operations.

Question 8. 20 165( points) Write the Javadoc for the insertAll() method in the CS232ArrayList class. Pay particular attention to writing clear documentation that will assist other programmers using this code. This homework question addresses the WiD (Writing in the Discipline) goal for our course, which is to write clear documentation for other programmers. In determining the style and level of detail of your comments, use the instructors’ comments in the homework assignments as a guide. Another good source of examples for the style of Javadoc comments is the official Java API. You should employ standard Javadoc formatting conventions.