CSC1016S Assignment 8: Linked Lists

Assignment Instructions

This assignment concerns solving problems involving reasoning about linked lists, constructing code that manipulates linked lists, and devising JUnits tests of linked lists.

Question 1

Write a program to load a text file into a list (of Strings) and print out its contents with each line prefixed by the line number and number of non-space characters in the line.

The name of the text file is supplied as the first command-line parameter.

You may use any kind of linked list (singly, doubly, etc.). You may use the standard Java class or write your own.

Sample File

```
this
is
just
a
test
```

Sample Output

```
1/4: this
2/2: is
3/4: just
4/1: a
5/4: test
```

Question 2

Write a program to load a text file into a linked list, sort the lines based on increasing length and then print them to the screen (using the formatting from Question 1).

The name of the text file is supplied as the first command-line parameter.

You may use any kind of linked list (singly, doubly, etc.). You may use the standard Java classes or write your own. You may use any sorting algorithm.

Sample File

```
this
is
just
a
test
```

Sample Output

```
1/1: a
2/2: is
3/4: this
```

CONTINUED

```
4/4: just
5/4: test
```

Question 3

On the Vula assignment page you will find a class called 'SimpleLinkedList'.

The class uses an inner class called 'Node', implements the Iterable interface, and has another inner class called NodeIterator that implements the Iterator interface.

Class SimpleLinkedList

A simple linked list implementation that uses an inner Node class.

Constructors

```
public SimpleLinkedList()
        // Create an empty list.

public SimpleLinkedList(T[] items)
        // Create a list containing the given items.
```

Methods

```
public T get(int index)
        // Obtain the item at the given index.

public void set(int index, T item)
        // Replace the item at the given index with the given item.

public void add(T item)
        // Add the given item to the end of the list.

public void addAll(T[] items)
        // Add the given items to the end of the list.

public void insert(int index, T item)
        // Inserts the given item at the given index.

public void remove(T item)
        // Removes the first occurrence of the given item from the list.

public void removeAt(int index)
        // Removes the item at the given index.

public Iterator<T> iterator()
        // Obtains an iterator for traversing the list.

public int indexOf(T item)
        // Returns the index of the first occurrence of the given item in the list.
        // Returns -1 if item not in the list.

public int size()
        // Obtains the size of the list.

public void trimToSize(int size)
        // Discard all elements beyond index (size-1).
```

CONTINUED

Write JUnit tests for SimpleLinkedList, putting them in a class called SimpleLinkedListTest (stored in the file `SimpleLinkedListTest.java)`. You must include the following 4 tests:

- check that a linked list created using the constructor (with a list of items passed in as a parameter) contains the items passed in.
- check that indexOf applied to search through a linked list of items returns the correct index for each item in the list.
- check that the insertion of an item in a linked list of items using insert (...) results in the item being inserted and appearing in the correct position.
- check that trimToSize (...) does truncate the list to the size specified.

Refer to the instructions in assignment 7 for a refresher on JUnit usage.

Marking and Submission

Submit `Question1.java`, Question2.java, and `SimpleLinkedListTest.java` in a single .ZIP folder to the automatic marker.

The zipped folder should have the following naming convention:

yourstudentnumber.zip

END