CSC171 — Homework 17

DIY Data Structures

The goal of this assignment is to give you experience with implementing your own data structures, particularly lists. While working on these programs, make sure you understand how to analyze the performance of your data structures in terms of their *worst-case behavior as a function of the number of elements in them*. Chapter 21 of the textbook is a good additional resource for this material.

Questions

1. Implement an `ArrayList` class that stores a list (of objects) using an array. Your class should have an `append` method to add an element to the list. Your `main` method should demonstrate the use of the class and method, on `Integer`s or instances of whatever class you like.

2. Add a `toString()` method that returns a String representation of an `ArrayList` and use it in your `main` method.

3. Add a `prepend` method to add an element at the front of an `ArrayList`

4. Make your class generic by adding a generic type parameter and using it appropriately throughout the code.

5. The following code is the start of the `LinkedList` class described in lecture.

   ```
   public class LinkedList {

       protected class Node {
           public Object data;
           public Node next;
       }

       protected Node first;

   }
   ```

   Add the `prepend` method that adds a new `Object` to the front of the list.

6. Add a method `indexOf(Object e)` that returns the index of the given object in the list, or -1 if the object is not in the list.

7. Add a method `get(int i)` that returns the object at the given index in the list, or throws an `IndexOutOfBoundsException` if there is no such element in the list.

8. Add an instance variable `last` and use it to implement an efficient (fast) `append` method that adds a new `Object` at the end of the list.
9. Make your class generic by adding a generic type parameter and using it appropriately throughout the code.

Grading Scheme

Equal weight for each part.

Doesn't compile or is trivial	$<$ 50%
Compiles and is non-trivial	$\geq$ 50%
Complete and correct with good style and comments	100%
Incomplete, incorrect, bad style, no comments	$<$ 100%

Submission Requirements

Your submission **MUST** include a file named “`README.txt`” with your name, your NetID, the assignment number, and your lab section. This file should explain anything we need to know about how to build and run your project. In particular, be sure to explain how to run what parts of your submission for each question in the assignment.

Submit your solution as a single ZIP archive to BlackBoard before the deadline.

Late homeworks will not be graded and will receive a grade of 0.

All assignments and activities associated with this course must be performed in accordance with the University of Rochester's Academic Honesty Policy.