CS1083 Assignment # 9 – Fall 2021

Due: Wednesday, November 24th before 11:00 pm (Atlantic time) in the Desire2Learn Assignment 9 submission folder. (See submission instructions below).

The purpose of this assignment is:

- Introduce you to working with linked lists, including using recursion

Preparation:

- Review the textbook examples of linked lists from Ch. 13.

This assignment is to be done individually. If you have questions, direct them to a tutor/assistant during a help session in the "Faculty of Computer Science Student Success Centre" team or to your course instructor

---

1. Linked List of Integers

You will need to download the zip file from D2L which contains the files referred to in this assignment.

The file IntList.java contains definitions for a linked list of integers. The class contains an inner class IntNode that holds information for a single node in the list (a node has a value and a reference to the next node) and the following IntList methods:

- public IntList()— constructor; creates an empty list of integers
- public void addToFront(int val) — takes an integer and puts it on the front of the list
- public void removeFirst() — removes the first value from the list
- public void print() — prints the elements in the list from first to last

The file IntListTest.java contains a driver that allows you to experiment with these methods. Save both of these files to your directory, compile and run IntListTest, and play around with it to see how it works.

Then add the following methods to the IntList class. For each, add an option to the driver to test it.

- public int length() — returns the number of elements in the list.
- public void addToEnd(int val) — takes an integer and puts it on the end of the list
- public void removeLast() — removes the last element of the list. If the list is empty, does nothing.
- public void replace(int oldVal, int newVal) — replaces all occurrences of oldVal in the list with newVal. Note that you can still use the old nodes; just replace the values stored in those nodes.

After you finish adding each new method, test your program by adding the method as an option in the IntListTest class.

2. Printing a List Backwards

Add the following methods to the IntList class. Then for each new method, add an option in the driver IntListTest.java to test it.

• public void printRec() — prints the list from first to last using recursion. Hint: The basic idea is that you print the first item in the list then do a recursive call to print the rest of the list. This means you need to keep track of what hasn't been printed yet (the "rest" of the list). In particular, your recursive method needs to know where the first item is. Note that printRec() has no parameter so you need to use a helper method that does most of the work. It should have a parameter that lets it know where the part of the list to be printed starts.

• public void printRecBackwards() — prints the list from last to first using recursion. Hint: Printing backward recursively is just like printing forward recursively except you print the rest of the list before printing this element.

After you finish adding the new methods, test your program. (Aim for good test coverage.)

Submission instructions are on the next page...

For this assignment, only an electronic submission is required.

Your electronic submission (submitted via Desire2Learn) will consist of two files:

i. a written report. This should begin with a title page. As always, your title page should include: the course (CS 1083), the assignment number (Assignment #9 in this case), your full name, and your UNB student number. That should be followed by each part clearly identified with a section heading. Include:

   a. the updated source code for the IntList class containing the new methods from Question 1& 2
   b. the updated source code for the IntListTest class including options for testing the new methods from Question 1 & 2
   c. the sample output generated from testing your program

   This written report should be prepared using a word processor; we recommend using Microsoft Word (i.e. create a .docx file for your report). Copy & paste your java source code & required output into the report document. Add appropriate headings for each part. Fix up the formatting where necessary, adjusting line breaks & page breaks to ensure that your document is easy to read. Use a monospaced font for your code to maintain proper indentation.) Once the report is complete and you've checked it all over, save the .docx file for your own records, and then **save a second copy in pdf format for submission**. (Note: Be sure to open that file in a pdf viewer to verify that the pdf was generated correctly.) The **SINGLE pdf file** containing your report will be submitted to the appropriate assignment submission folder on Desire2Learn. (It is important that you submit a pdf file and NOT the original Word document. This pdf will allow the marker to write comments directly on your work to give you better feedback.)

   Note: Please name this report as follows: **YourName_A9_Report.pdf**

ii. an archive file (.zip) that contains all your work for this assignment. Make sure that your archive includes **all source code** (.java files - in case the marker wishes to compile & run your code) and sample output. This archive should be submitted as a single file to the appropriate submission folder on Desire2Learn. Note: Please name this archive file as follows: **YourName_A9_Archive.zip**