CS163 Review – Linear Linked Lists

This worksheet is required based on proficiency demonstration scores

*It is **optional** if you receive an **E** (Excellent) or **P** (Proficient) on the midterm proficiency demos*

Submit code to the CS199 D2L dropbox. Limit the time invested to 1 hour and 50 minutes maximum.

Coding: *With this review lab, we will be working with an existing linear linked list of integers on unix. You will have access to the class interface in a .h file available on D2L's online "locker" which you may examine. Your job will be to implement functions to review your linear linked lists.*

_____Step 1. Write a function to display all items in the linear linked list that is supplied

a. Prototype: void list::display_all();
b. Create a current pointer defined to assist with traversal
c. Watch your loop's stopping condition. Did all of the data get displayed?
d. Add a function call in main.cpp
e. Download the .h and .o files from D2L's online "locker"
f. Compile: g++ *.cpp *.o
g. Run: ./a.out

______Step 2. Write a function to count the number of times the first number (in head's data) appears in the list.

a. Prototype: int list::count_first();
b. Check to make sure you compare the last node's data as well
c. Add a function call to main.cpp
d. Compile: g++ *.cpp *.o
e. Run: ./a.out

_____Step 3. Design the code to find out if the LAST number appears more than once

a. **First: How many temporary pointers do we need?**_______________________
b. **Next: On paper, write the loop to FIND the last node:** *Make sure to watch when you stop traversal*…circle the correct answer:
   while (current) or while(current->next)

c. **Lastly: On paper, re-traverse (starting at head) comparing the data:**

_____Step 4. **Experience the mechanics of appending:**

a. When should we stop traversal? <u>while (current)</u> or <u>while(current->next)</u>

b. Write a loop to traverse **to** the last node:

c. Write the code to attach a new node, connecting it to this last node:

d. Set the next pointer for this "new" last node to NULL:

e. *Based on your code, draw the pointer diagram here:*

_____Step 5. Implement the code now online to add a node to the end of a linear linked list:

a. Prototype: void list::append();

b. Call the function from main

c. Compile: g++ *.cpp *.o

d. Run: ./a.out

_____Step 6. Implement the code to remove the last item from a linear linked list:

a. Prototype: void list::remove_last();

b. Call the function from main

c. Compile: g++ *.cpp *.o

Self-Assessment: *What could you do to improve for the final proficiency demo?*