COSC 251 – Programming Languages
Project 1
Spring 2016

Objective: Implement a data structure in C/C++ that is easily dealt with in JAVA.

Your Task: You will create an Object Oriented version of the linked list data structure that we will review, briefly, in class (you all should have seen it in your 201 class). You will code this as a doubly-linked list and will have to provide the following functionality:

Addition of a node to the start of the list.
Addition of a node to the end of the list.
Addition of a node in an arbitrary spot inside of the endpoints.
Remove a node at the start of the list.
Remove a node at the end of the list.
Remove a node in an arbitrary spot inside of the endpoints.
Sort the list by id number (see below) in ascending order.
Clear the list.
Return/Get the node at an arbitrary index.
Print out the entire list.

This functionality will be tested through execution of various methods that are specified below. This project will test your understanding of pointers and pointer chasing. Each node should be its own object and contain an instantiation of a class called Student which has three private members (int id, double gpa, String name) and get/set functions as well as a constructor. The node class should also contain the pointers that you'll need to complete the functionality of the linked list.

Your LinkedList class should hold your actual linked list made of nodes and contain the functionality noted in the list above. This will be the primary class that our driver will interface with. Also note, since your LinkedList is a doubly-linked list, we expect that any method that has to traverse the LinkedList traverses from the closest end.

Code Requirements:

Your project files must follow the requirements below. If they do not, it will either not work with our driver, or it will not be structured correctly for a LinkedList. You must implement the following methods with the exact method signature noted:

In LinkedList.h/LinkedList.cpp

```
LinkedList() //constructor call
int length() //returns length of the LinkedList
void printList() //prints every member of the LinkedList
```

```
void add(int i, Student element) //adds element to the LinkedList at index i
Student front() //returns the Student at index 0
Student back() //returns the Student at index length-1
Student get(int i) //returns the Student at index, should return a default constructed
                   //Student if i is not a valid index
Student remove(int i) //remove the student at index i and return, should return a default
                      //constructed Student if i is not a valid index
void sort() //sorts the LinkedList by Student id
void clear() //empties the list
```

In Node.h/Node.cpp

```
Node(Student s) //Constructor call
Student getStudent()
```

In Student.h/Student.cpp

```
Student(int i, double g, string n) //Constructor call, sets id, gpa, and name
int getID()
double getGPA()
string getName()
```

You may have more methods and classes than specified, but not less.

Testing:

You should test your code against our driver (posted in the next week). If your code does not complete correctly (i.e. the correct, matching outputs) with our driver, then you have code issues that you must fix. If your code works correctly with our driver then you can be reasonably assured that you will do well. Note that errors will need to be caught and dealt with without crashing the driver program. See the driver program for specifics as all errors we will be testing for are tested in the driver.

Deliverables: the class files for Student, Node, and LinkedList, plus any other class files you may need.

Expectations: The code should be clean, concise, well-commented and correct. If you use an outside source, be sure to document that source. Significant use of outside sources will result in a deduction. Grading rubric and example binary will be provided a week ahead of the due date. You may work in pairs on this project and pairs across sections are acceptable. If you are going to work in a pair, you must email the appropriate professor (or professors if you are working with someone from the other section) with your team by 5pm on Tuesday, February 9th. Failure to email by that deadline will mean that you will be working alone for this project.

DUE: February 19th, 11:59pm via Blackboard