COSC 251 – Programming Languages
Project 1
Spring 2017

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 stack data structure that we will review, briefly, in class (you all should have seen it in your 201 class). Internally, you may code this as whatever data representation you choose, but will have to provide the following functionality:

push, pop, and top functionalities
Sort the stack by id number (see below) in ascending order.
Clear the list.
Print out the entire stack.
Boolean check for empty stack.
Size return.

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 stack.

Your Stack class should hold your actual stack 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 Stack is a stack, we will not be allowing for arbitrary index gets.

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 Stack. You must implement the following methods with the exact method signature noted:

In Stack.h/Stack.cpp

```
Stack() //constructor call
int length() //returns length of the Stack
void printStack() //prints every member of the Stack, in proper order
void push(Student element) //adds element to the Stack
Student top() //returns the Student at index 0
```

void pop() //remove the student on top
void sort() //sorts the stack by Student id
void clear() //empties the stack
Student topandpop() //removes and returns the top element of the stack
boolean isEmpty() //returns true if the stack is empty
int size() //returns the size of the stack

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 Stack, 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 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 me with your team by 5pm on Tuesday, February 7th. Failure to email by that deadline will mean that you will be working alone for this project.

DUE: February 17th, 11:59pm via Blackboard