CST8234- C Language

Assignment 2

Problem statement:

In this assignment, you will work on building a linked list implementation. You will implement a linked list that stores a student struct, which holds student information.

This assignment is intended to get you to practice using pointers, linked list data structure, header files, and user inputs.

Background Information:

This assignment is designed to help understand what linked lists are, how they work internally, and how to use them. You will implement the common functionality of a linked list and then use one to build a list of student structs.

Each student is represented by their first and last name (i.e. a struct of 2 fields, each char[20]). Then, each student is linked to the next student to create a linked list of students.

To encapsulate the data, which is the student struct in our case, we will create a *node* struct that will hold the student struct and a reference to the next student struct in the list.

The program will ask the user to enter each student information then proceed to build a linked list of students based on the user entry.

Requirements:

Write a program that achieves the following requirements:

1. Implement all functions defined in **Node.h** file in a separate 'C' file (you should find the slides very helpful to complete the implementation).
2. The program instructs the user on the actions they will perform while it's running all the time. (In other words, make sure you have meaningful and readable printf( ) statements).
3. The program should have a structure that is called Student,and type defined as **student_t**, with the following information:
   1. First name (a 'C' string of 20 characters including the null character ('\0')
   2. Last name (a 'C' string of 20 characters including the null character ('\0')
4. The program should have a structure that is called Node, and type defined as **node_t**, which has the following information:
   1. A student_t pointer, which represents the *value* of the node.
   2. A pointer to the next Node in the list (which is a student; see previous bullet item).
5. The program should have a pointer to a linked list called head, which is of type node_t.
6. The program should read the first 3 students from the user and add them to the list by adding to the beginning of the list.
7. Then, the program will read another 3 students' information but will add them to the end of the list.
8. The program then will delete the first 3 elements in the list.
9. Then the program will delete the last 3 elements in the list; the linked list is empty.
10. The program will read 3 new students' information from the user and add them to the end of the list.
11. Finally, the program will delete the second element in the list only, keeping the first and last in the list.

Design Requirements:

1. You are given a header file, called 'Node.h' which contains the prototypes of the functions you need to implement. **DO NOT** change any of the function prototypes.
2. Add in the two (2) struct definitions in the header file so that you can use them everywhere.
3. Create a new '.c' file called 'Node.c' to have the full implementation of the defined functions in it.
4. In a separate file, declare the main function and only include the header file, and not the 'C' source file.
5. Implement the functionality described in the requirements section.
6. **YOU MUST** build your code using all build flags mentioned in the slides as '-w', '-Wall', '-pedantic' and '-ansi' switches. Failing to do so will make you lose points on the assignment.
7. Make your source code readable and format it using eclipse IDE: Source > Format

Documentation Requirements:

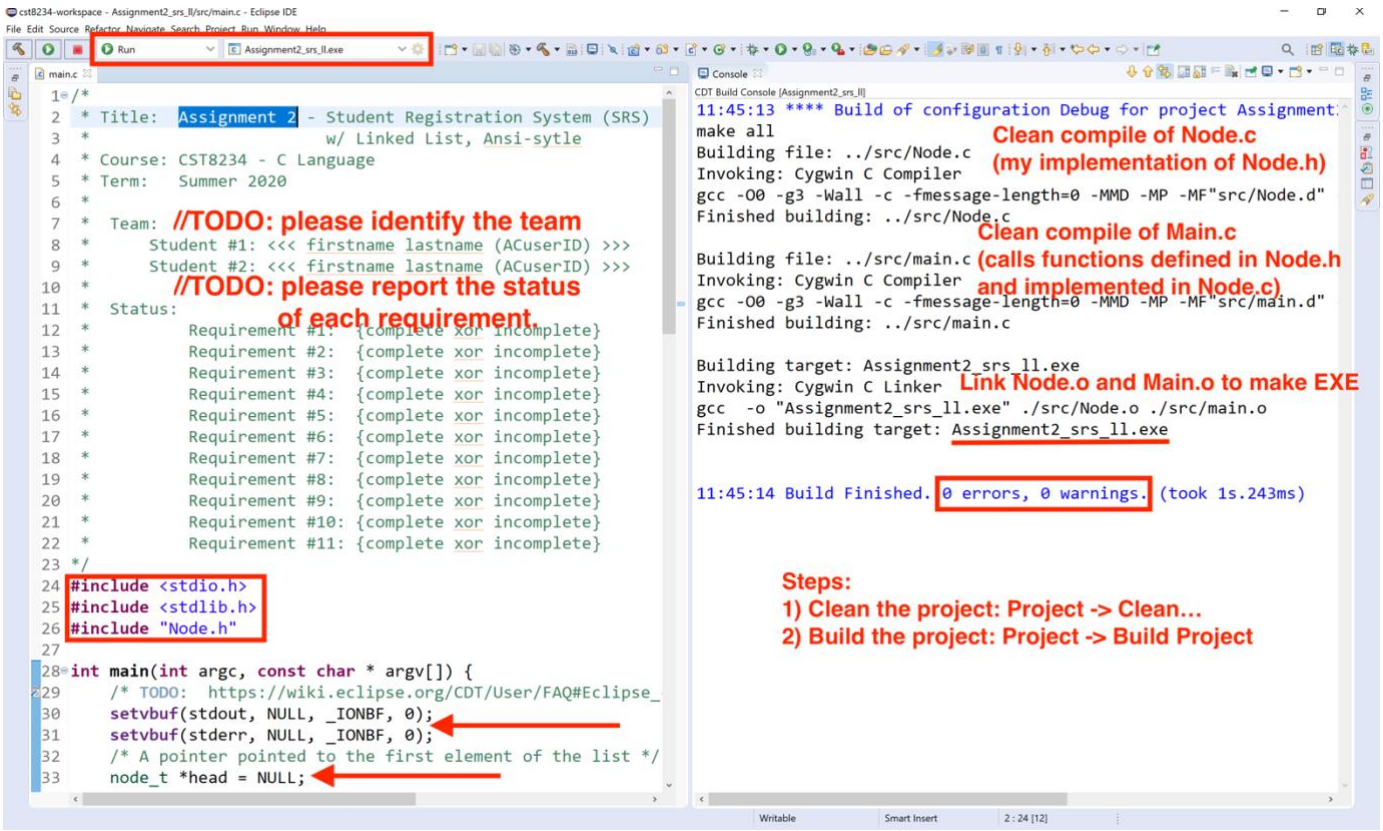
Document your team's solution by adding a header comment to the start of your C source file that has the main( ) function:

```
/*
 * Title:  Assignment 2 - Student Registration System
 *                        w/ Linked List, Ansi-sytle
 * Course: CST8234 - C Language
 * Term:   Summer 2020
 *
 *  Team:
 *  Student #1: <<< firstname lastname (ACuserID) >>>
 *  Student #2: <<< firstname lastname (ACuserID) >>>
 *
 *  Status:
 *          Requirement #1:  {complete xor incomplete}
 *          Requirement #2:  {complete xor incomplete}
 *          Requirement #3:  {complete xor incomplete}
 *          Requirement #4:  {complete xor incomplete}
 *          Requirement #5:  {complete xor incomplete}
 *          Requirement #6:  {complete xor incomplete}
 *          Requirement #7:  {complete xor incomplete}
 *          Requirement #8:  {complete xor incomplete}
 *          Requirement #9:  {complete xor incomplete}
 *          Requirement #10: {complete xor incomplete}
 *          Requirement #11: {complete xor incomplete}
*/
```

Reference Screenshot #1:

Your team's project must compile without warnings and without errors.

Compare your screenshot to the reference screenshot:
Notice the //TODOs

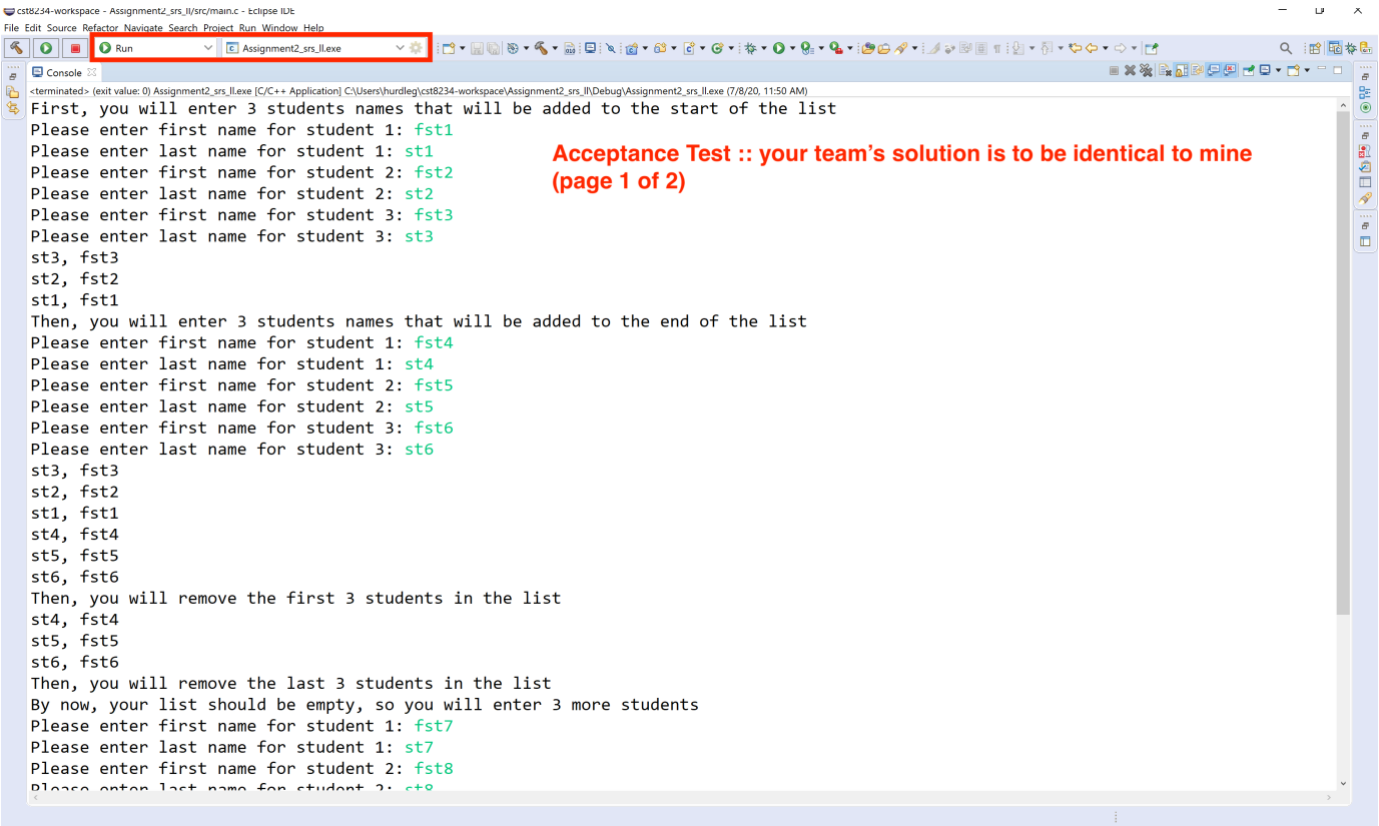


Reference Screenshot #2:

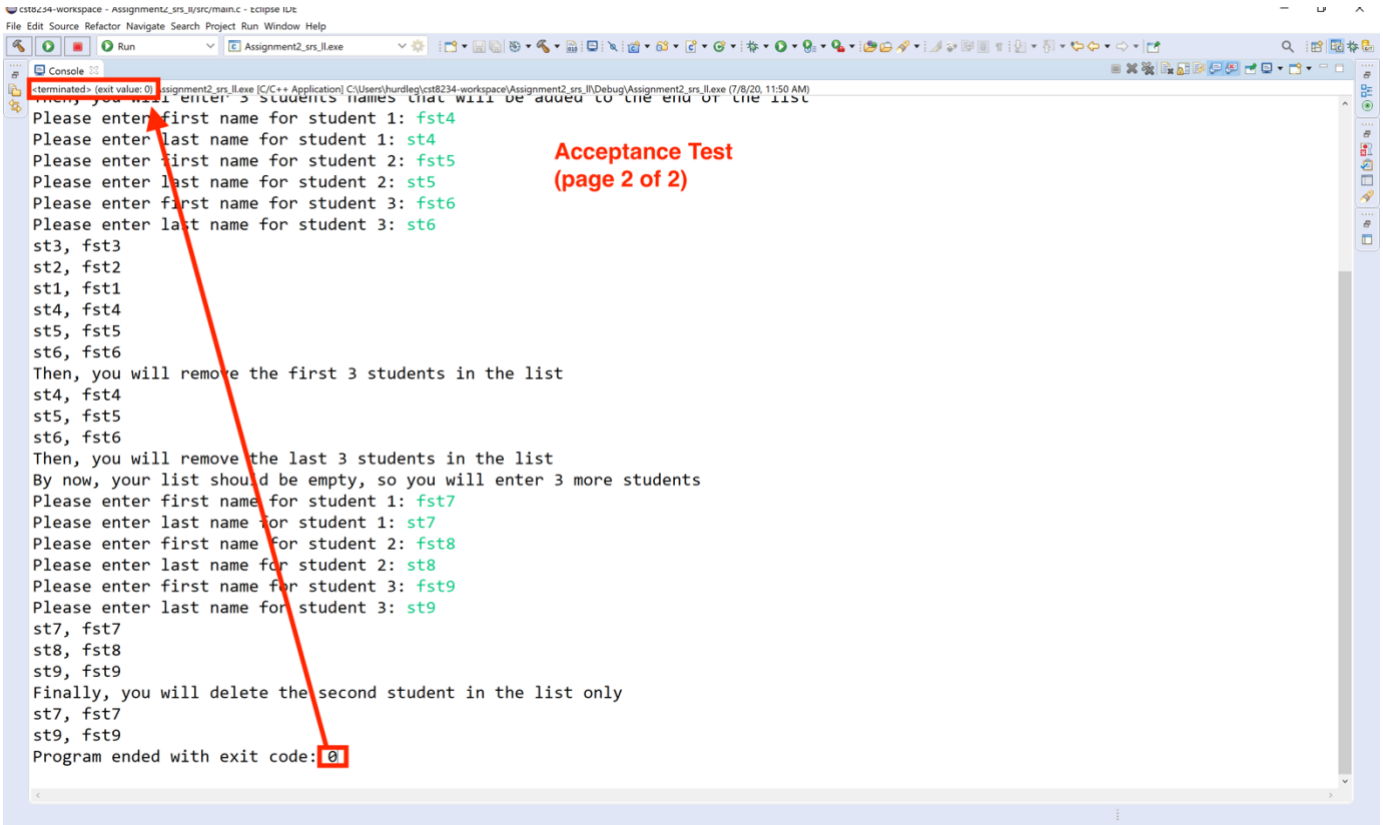
Demonstrate your team's solution implements the functional requirements.

Compare your screenshot to the reference screenshots:

Screenshot #1 of 2:



Screenshot #2 of 2:



Supporting Files:

Along with this document, you will find a header file included in Brightspace. The file is named **Node.h**. These files contain prototypes for functions that you will implement. Do not alter Node.h: do not add anything to it; do not delete anything from it; do not alter anything written in it; *Look, don't touch*.

Submission instructions:

1. Late submissions are **not** accepted. Why? Us faculty have due dates too. Our department, ICT-AP, requires faculty to submit their course Final Grades on-time so that you, the student, can see your official Final Grades, as released by the Registrar's Office (RO). We appreciate your understanding.
2. This assignment is to be completed in teams of size two (2). That is, you and a partner will collaborate on this assignment. Your partner can be the same or different from Assignment 1. Your partner can be in a different lab section. Please choose your partner wisely as squabbles will be solved by dividing the mark in half (i.e. ½). Individual submissions will not be accepted.
3. From eclipse IDE, export your Assignment 2 project as an archive file. Refer to eclipse documentation for details on how to export:
    - https://dzone.com/articles/exporting-and-importing
4. Make a zip-file that contains the following items:
    - eclipse IDE archive file
    - screenshot #1: clean compilation
    - screenshot(s) #2 (& #3): acceptance test
5. Name your zip-file to include your AC userID and your partner's AC userID. Follow this format:
   cst8234_assignment_2_yourACUserID_yourPartner'sACUserID
   For example: cst8234_assignment_2_bond0007_jaws0001.zip
6. Partner #1: upload and submit the zip-file to Brightspace before the due date.
7. Partner #2: repeat the previous step for yourself.