COSC2430: Programming and Data Structures

University roster using linked lists

Introduction

In this homework, you will create a C++ program to manage the list of enrolled students in a university. The data of each student will be stored as nodes of an **ordered linked list.** You should be able to add and delete students from the list, and perform other operation on the stored data.

Student information include:

- ID number (integer of 5 digits)
- First name
- Last name
- Major
- GPA (double, between 0 and 4)
- Current Credit Enrolled (integer, between 0 and 15)

Input and Output

The input is a single text file. Each file is divided in blocks that call different operations on the linked list, followed by the necessary arguments. The name used to declare the instruction to execute will be all in upper cases. Each new argument is in a new line of the file. Your program should go through the file and execute all the listed operations, and terminate when the end of file is reached.

It is fair to assume that all calls will be in upper case letters and will always be separated by an empty line, even when the number of arguments is incorrect for the function call. The file terminates with new line and EOF.

Example 1 of input

```
INSERT
03948
Tom
Brady
Physics
3.40
12

INSERT
03491
Joe
Flacco
Chemistry
```

3.91
12

PRINT_ROSTER

PRINT_STUDENT
Joe
Flacco
<EOF>

The output of each function call should be appended to an output file (name specified in the command line).

Example 1 of output

Tom Brady, 03948
Joe Flacco, 03491

Joe Flacco, 03491
Major: Chemistry
GPA: 3.91
Credits enrolled: 12

Complete list of instruction calls for ordered linked list

Call	Description	Arguments	Output
INSERT	Insert a new student in the linked list ORDERED BY LAST NAME. Duplicates should be rejected (by ID or first + last name)	ID_number First Name Last Name Major GPA Current Credits	-
PRINT_ROSTER	Print first name, last name and ID of all students	-	<FirstN> <LastN>, <ID> …
PRINT_BY_MAJOR	Print first name, last name and ID of all students in a major	Major	<FirstN> <LastN>, <ID> …
PRINT_BY_GPA	Print first name, last name and ID of all students with GPA greater or equal to argument	GPA	<FirstN> <LastN>, <ID> …
PRINT_STUDENT	Print all data of student	First Name Last Name	<FirstN> <LastN>, <ID> Major: <Major> GPA: <GPA> Credits Enrolled: <CurrentCE>

DELETE_STUDENT	Delete student from list based on first and last name	First Name Last Name	-
DELETE_ID	Delete student from list based on ID number	ID number	-
UPDATE_GPA	Set GPA of student to new value	First Name Last Name GPA	-
UPDATE_MAJOR	Change student major to new value	First Name Last Name Major	-
ADD_CLASS	Add class for student (reflects on credits enrolled)	First Name Last Name credits	-
REMOVE_CLASS	Remove class for student (reflects on credits enrolled)	First Name Last Name credits	-
GPA	Compute average GPA across students	-	GPA mean: <average_GPA>

The main C++ program will become the executable to be tested by the TAs. The result should be written on another text file (output file), provided on the command line. The input and output files are specified in the command line, not inside the C++ code.

The general call to the executable (sum_rowcol, in this example) is as follows:

```
uni_roster "A=<file>;C=<file>"
```

Call example with one input file and another output file.

```
uni_roster "A=a.txt;C=c.out"
```

Requirements

- Your ordered linked list **must inherit from the linkedListType ADT** presented in class and available in the book (pg. 278-292).
- **It is NOT allowed to use vector classes or other classes provided in the STL.**
- Your C++ code must be clear, indented and commented.
- Each function call should check for input correctness. If the input is of invalid format, do not perform the operation and print an error on screen. Examples of input correctness include right number and type of arguments, or checking for argument boundaries (for GPA, credits and ID).
- GPA must be printed with 2 decimal points.

- Your program will be tested with GNU C++. Therefore, you are encouraged to work on Unix, or at least make sure that your code compiles and runs successfully on the server.
- You can use other C++ compilers, but the TAs cannot provide support or test your programs with other compilers.
- The output file must contain the result in the format specified in the table. A single empty line should separate each output segment.

Testing for grading:

- Your main cpp program will be automatically compiled. If there is an error in compilation it will not be executed. **Programs that do not compile on the server will receive a score of 0.** You are responsible of the content of your submission folder.
- **The name of your submission folder must be hw2**. The folder must not be nested in another folder.
- Submitted files must be limited to headers and source files (.h and .cpp).
- Your program should not crash, halt unexpectedly, take an unusual amount of time to run (more than 10 seconds) or produce unhandled exceptions.
- Your program will be tested with 10 test cases, going from easy to difficult.

DEADLINE: October 3rd, 5:00PM – NO LATE SUBMISSIONS ALLOWED

In general the due date cannot be changed. No exceptions, unless there are medical or exceptional reasons.

Grading

- A program not submitted by the deadline is zero points.
- A program that compiles but does not work program is worth 10 points.
- Failing to derive the linked list from the linkedListType ADT will result in a 10 points penalty.
- A code with no comments will receive a 5 points penalty.
- **Bonus 5 points:** if your code can handle the call INSERT with missing arguments for one or more among IDnumber, GPA, Major and Current Credits, filling the blanks with default values (0 for numbers and “” for major). You can assume you will always have first and last name of the student.

Plagiarism and cheating: C++ code is individual: it cannot be shared (other than small fragments used in class or in lab examples). Programs will be compared for plagiarism. If the TAs detect that a portion of your C++ code is highly similar to C++ on the Internet or other student the grade will be decreased at least 50%.