

Introduction to C Programming II

Summary: In this homework, you will be implement a simple program that manages a course schedule.

1 Background

In this assignment, you will write a program that manages a set of courses a student is taking. This program will introduce new keywords (struct, enum), and the handling of heap memory allocations. Similar to the last assignment, we ask that you do your program development on Linux.

The program will store a collection of courses, giving the user options to add to the collection or remove from that collection. The collection of courses will be stored as a linked list (using structs as nodes). The problem must support any number of courses in the collection. Courses will be composed of a subject (enum), number, teacher, and the number of credits. When a user adds a course, a new node must be created, and when a course is removed, its corresponding node is removed. When the program exits, it will save the courses that have been added to a text file. When it starts, it will check if there is course data to load, and load any existing data.

This document is separated into four sections: Background, Requirements, Compiling a C Program from Terminal, and Submission. You have almost finished reading the Background section already. In Requirements, we will discuss what is expected of you in this homework. In Include Files, we list several files that may provide useful functions for the assignment. Lastly, Submission discusses how your source code should be submitted on Canvas.

2 Requirements [30 points]

For this assignment, you will write a course scheduler program in C. Your program **must**:

- **Compile on Xubuntu (or another variant of Ubuntu) 22.04.**
- Have an enumeration type (called Subject) to represent subjects (SER, EGR, CSE, EEE).
- Have a struct type (called CourseNode) to hold information about a course. Include a subject (see enum type above), a number (int), a teacher (a string of length 1024), and credit hours (int). It should also include the variables necessary to represent a linked list.
- Use a linked list of structs (called course_collection) to store course information. Your program will allow the user to enter any number of courses - this means you must use dynamic memory allocation to store the courses. When a user adds a course, a new node must be created, and when a course is removed, its corresponding node is removed.

Note: your program may not exactly match the outputs below - screenshots are only provided as samples. Points allocation is:

- Update the main menu to display the total number of credits each time it is shown. [2 points]

```

Terminal - ruben@ruben-VirtualBox: ~/Desktop
File Edit View Terminal Tabs Help
ruben@ruben-VirtualBox:~/Desktop$ ./scheduler

Welcome to ASU Class Schedule

Menu Options
-----
a: Add a class
d: Drop a class
s: Show your classes
q: Quit

Total Credits: 0

Please enter a choice --->

```

- The program must not leak memory on exit. [4 points]
- Create a void `course_insert()` function that adds courses to the collection and keeps it sorted. Order by both subject and course number. For example, all CSE courses would show before SER courses, and SER222 would show before SER334. Populate the elements of the `CourseNode` as shown below. Do not allow duplicate courses. Two example inserts are shown. Notice that the integer input for different courses is shown to the user to help them select (you may number them differently). [8 points]

```

Terminal - ruben@ruben-VirtualBox: ~/Desktop
File Edit View Terminal Tabs Help
Please enter a choice ---> a
What is the subject? (SER=0, EGR=1, CSE=2, EEE=3)?
0
What is the number (e.g. 240)?
222
How many credits is the class (e.g. 3)?
3
What is the name of the teacher?
Acuna
Menu Options
-----
a: Add a class
d: Drop a class
s: Show your classes
q: Quit

Total Credits: 3

Please enter a choice --->

```

```

Terminal - ruben@ruben-VirtualBox: ~/Desktop
File Edit View Terminal Tabs Help
Please enter a choice ---> a
What is the subject? (SER=0, EGR=1, CSE=2, EEE=3)?
2
What is the number (e.g. 240)?
230
How many credits is the class (e.g. 3)?
3
What is the name of the teacher?
Chen
Menu Options
-----
a: Add a class
d: Drop a class
s: Show your classes
q: Quit

Total Credits: 6

Please enter a choice --->

```

- Create a void `schedule_print()` function. This function will display the contents of the `course_collection` list with format Subject Number Credits Teacher as shown. (Hint: Use a switch statement to map from an enumeration to a printf.) [2 points]

```

Terminal - ruben@ruben-VirtualBox: ~/Desktop
File Edit View Terminal Tabs Help
a: Add a class
d: Drop a class
s: Show your classes
q: Quit

Total Credits: 6

Please enter a choice ----> s

Class Schedule:
CSE230 3 Chen
SER222 3 Acuna

Menu Options
-----
a: Add a class
d: Drop a class
s: Show your classes
q: Quit

Total Credits: 6

Please enter a choice ---->

```

- Create a void `course_drop()` function. This function will prompt the user for a course by its subject and number. It then searches for the course in `course_collection` and removes it. Dropping a course is only allowed if the course exists in the collection, otherwise an error message will be displayed. [4 points]

```

Terminal - ruben@ruben-VirtualBox: ~/Desktop
File Edit View Terminal Tabs Help
d: Drop a class
s: Show your classes
q: Quit

Total Credits: 6

Please enter a choice ----> d

What is the subject?
SER

Enter number:
222

Menu Options
-----
a: Add a class
d: Drop a class
s: Show your classes
q: Quit

Total Credits: 3

Please enter a choice ---->

```

```

Terminal - ruben@ruben-VirtualBox: ~/Desktop
File Edit View Terminal Tabs Help
a: Add a class
d: Drop a class
s: Show your classes
q: Quit

Total Credits: 3

Please enter a choice ----> s

Class Schedule:
CSE230 3 Chen

Menu Options
-----
a: Add a class
d: Drop a class
s: Show your classes
q: Quit

Total Credits: 3

Please enter a choice ---->

```

- Properly deallocate memory when removing nodes. No memory leaks. [2 points]
- Create a void `schedule_load()` function that will be called when your program starts. This function should check if a data file exists (plain text, filename up to you), and load any courses that it specifies. Courses should be sorted once the program finishes loading. It may assume that the file is in the format used by your `schedule_save()` function. If the file does not exist, it should do nothing. [4 points]
- Create a void `schedule_save()` function that will be called when your program ends. This function saves the contents of the `CourseCollection` to a plain text file (filename up to you). If the file already exists, you may override it. You should design the format of the data file yourself. [4 points]

3 Include Files

Throughout the assignments in SER334, you will need to explore and learn various libraries. To complete this assignment, you may find the following include files and functions useful:

- *stdio.h*: Defines standard IO features, including file routines.
 - Useful types:
 - * *struct FILE* - A structure to represent a file handler.
 - Useful functions:
 - * *FILE** *fopen* (*const char** filename, *const char** mode): the *fopen()* function opens a file. It takes a filename, and mode, and returns a *FILE* value representing a file handler. Mode may be “w” for text write, or “r” for text read. If the file cannot be opened, it will return a NULL pointer.
 - * *char** *fgets*(*char** str, *int* num, *FILE** stream): the *fgets()* function reads a number of characters into a string variable (or stops early if it sees a new line) from a file handler. Typically used with a fixed size buffer variable.
 - * *int* *fprintf*(*FILE** stream, *const char** format, ...): the *fprintf()* function writes a formatted string into a file. Format section uses the same control sequences as *printf*.
 - * *int* *atoi* (*const char** str): the *atoi()* function takes a string and parses it into an integer that it returns.
- *stdlib.h*: Defines memory allocation functions.
- *string.h*: Defines string manipulation functions.
 - *char** *strcpy*(*char** dest, *const char** src): The *strcpy()* function copies the string pointed to, by src to dest.
 - *size_t* *strlen*(*const char** str): The *strlen()* function computes the length of the string str up to, but not including the terminating null character.

Your solution may not include all of these include files or functions, they are only suggestions. If you want to include any other files, please check with the instructor or TA before doing so. (Somewhat ironically a good documentation site for these methods is: <http://www.cplusplus.com/reference/cstdio/>.)

4 Submission

The submission for this assignment has one part: a source code submission. The file should be attached to the homework submission link on Canvas.

Writeup: For this assignment, no write up is required.

Source Code: Please name your source file as "LastNameScheduler.c" (e.g. "AcunaScheduler.c").