60-141 – Introduction to Programming II Winter, 2017
Assignment 4
(Saturday, March 25, 2017: 23:59:59)

Strings and Structures:

Write a C program that uses string processing functions to properly format a Comma Separated Values (csv) file of course Information and store them into an array of structures. Once all data have been formatted and stored, your program should be able to:

a. Add new course
b. Search for course information using either course name or course number as the search key
c. Display all or any course data.

The content of the input file (input.csv) is read by the program using input redirection technique that you already have used in the earlier assignments.

Assume that the CourseInfo structure has been defined as follows:

```
struct CourseInfo{
    int courseID;
    char courseName[50];
    char courseCode[12];
    char Term [6];
};
```

typedef struct CourseInfo courseInfo; //optional

Example:

Fields in the input file are comma separated as follows:
CourseName,Faculty,Subject,Level,Section,Semester,Year

Content of a sample input file:

```
programming,03,60,141,01,W,2015
programming,03,60,141,30,W,2015
Algebra,03,62,102,02,S,2013
Religion,08,98,938, 20,F,2014
```

Corresponding field values in the array of structures (after formatting):

CourseID	1	2	3	4
CourseName	PROGRAMMING	PROGRAMMING	ALGEBRA	RELIGION
CourseCode	0360-141-01	0360-141-30	0362-102-02	0898-938-20
Term	W2015	W2015	S2013	F2014

Hints:

1- For formatting data, the following points are to be considered:

a. Course ID should be unique sequential integers starting from 1 (<u>not</u> from 0).
b. Course name must be all capital letters (upper case).
c. Course Code should also be unique and store in the format as xxxx-xxx-xx. The first xxxx part indicates faculty code (xx) and subject code (xx). The second xxx part indicates level and the last xx part indicates section number. For example 0360-141-30
d. Term should be one character indicating semester (‘S’, ‘F’,’W’) following by 4 digits indicating a year. For example S2010, W2014.

2- Your program should implement at least the following functions:

a) **loadCourseInfo()** : To read all data from input file and format and store them in an array of structures.
b) **displayCourseInfo()**, To print a table indicating the course Information. For example:

```
ID      Name              Code         Term
1       PROGRAMMING       0360-141-01  W2015
2       PROGRAMMING       0360-141-30  W2015
3       ALGEBRA           0362-102-02  S2013
```

c) **addCourseInfo()**: To add a new course. Note that:
   - CouseID should be unique and generated automatically by your program (last courseID+1).
   - CourseCode is also unique and you cannot have two courses with the same courseCode. So, before adding a course, search for the courseCode to be sure that you have not had it previously.

d) **searchCourseInfo()**: To search for a course information using courseID or course Name and print the course information if it exists.

Requirements:

- Write and document a complete C program that is capable of satisfying the requirements of this assignment problem.
- UNDOCUMENTED OR POORLY DOCUMENTED code will automatically lose 50% marks.
- PLAGIARIZED work will not be graded and receive a mark of ZERO and reported according to the Senate bylaws.
- The question can use I/O redirection if appropriate. Please review the textbook for an example on using I/O redirection from flat files.
- TO SUBMIT: No later than the submission deadline, your assignment should be uploaded in Blackboard. Late submissions are not accepted and will receive a mark of ZERO.
- Upload your work as an attachment, include both the source file (assign4.c) and the script file (assign4.txt) - see below how to create the script file.

To create a script file (one that logs your compilation steps and your output in a text file):

1. script assign4.txt
2. cat assign4.c
3. cat input.txt
4. cc assign4.c
5. a.out < input.txt
6. ls -l
7. exit (DO NOT FORGET THIS STEP!!)

The example script execution presumes that the input was specified in the file input.txt. This may be changed if the input is provided interactively within the program, or if input is provided by using file I/O directly within the program.

If you are compiling your code under Cygwin shell, then you need to change line 5 to:

5. ./a.exe < input.txt

In line 4, students may prefer to use the gcc compiler to ensure improved warning and error diagnostic reports from compiling the program. You must have access to gcc in order to make this change, however.

NOTE: Submissions that are not received correctly by the deadline will automatically receive a ZERO mark. **Late assignment submissions are not accepted!**

NOTES:

1. Your assignment must be RECEIVED by the due date and time. Late assignment submissions are NOT accepted. Keep your script file, and all your code unmodified as proof of its completion in case it is not received.
2. It is your responsibility to get an early start on the assignment, research and ask questions ahead of time from the due date.
3. Marks will be deducted for unclear code (improper spacing and alignment, hard to read programs and missing outputs).
4. Make sure you turn in a complete script file that clearly shows: your code, your compilation process, a listing of the directory showing your source file(s) and the a.out with the date/time stamps, and the output. DO NOT SUBMIT a.out FILES!

5. **PLAGIARISM:** CHEATING IS NOT TOLERATED – PLAGIARISM IS CHEATING! You must submit your own work. Students who are suspected of copying someone else's work will be reported to the department's chair and the Dean of Science and be dealt with in accordance with the University policies. You should not share your code with others. Codes that are similar to each other will BOTH be reported as potential evidence of copying. It is imperative that you write your own code.

6. Authorized/limited help on this assignment may be provided directly from your Lecture or Lab instructors and Teaching Assistants.