Library System Project

Objectives

- Learn more about Structure
- Learn more about Files
- Practice on modeling a real-life example into programming
- Practice relation between structures
- Practice searching and sorting

Description

It is required to write a program that will maintain a library system. The system manages books information, library visitors, borrowing, ... etc.
The system stores for each **book** the title, author, publisher, ISBN, date of publication, number of copies, current available number of copies, and category.
In a library, a member may borrow many books at the same time. However, he must return the books before their due date. A **member** has a name (last and first), ID, address (building, street and city), phone number, age, and email address.
You should maintain a structure that links a user with the book he **borrowed**, you should store book ISBN, user ID, date borrowed, data due to return, and date returned.
You should have 3 separated files, one for books, another for users, and the last one for borrowing. It is better to read the files at the start of the program only, and you should save the changes only when the user selects save from the menu.

Tasks

Book Management

1. **Insert a new book**

This command adds a new book to our library. The added data is stored in a file. The file is a text file that is comma delimited in which each entry (book) is found on a separate line. Each entry must contain the main book information mentioned before. An example of one line in the file is as follows:

C How to Program, Paul Deitel, Pearson Education. Inc, 0-13-612356-2, 29/10/2009, 5, 3, Programming

2. Search for a book

The system should process a request by the user to look up information about a specific entry. The user can supply one of the following; book title (or part of it), author name, ISBN or category. The system should provide a list of books that matches the user request.

3. Add new copy

The system should prompt the user to enter book ISBN and the number of its copies and update the Book entry in the file with the new number of copies. (Validate it is not a negative number)

4. Delete book

The user should be prompted for the book ISBN and that entry should be deleted from the system (books file).

Member Transactions

5. Registration

Adding new user to our system. Data is stored in a separate file in the same format of books file (comma delimited, and each entry, user, is on a separate line).

Hamdy, Ahmed, 95314, 35, Shatby, Alexandria, 0123456789, 26, ahmed@gmail.com

6. Borrowing

For each member who borrows a book, we store book ISBN, user ID, date issued, date due to return, and date returned. Again, all borrowing data should be stored into a separate file. Also, the number of available copies for that book should be decremented. A Member cannot borrow more than 3 books at a time.

0-13-612356-2, 95314, 17/11/2017, 22/11/2017, null

7. Returning book

When a user returns a borrowed book back to the library, the entry for his borrowing action must be updated with the date returned. Also for book entry; the number of available copies of the book must be incremented.
The above record “in task 6” should be updated to look like:

0-13-612356-2, 95314, 17/11/2017, 22/11/2017, 21/11/2017

8. Remove member

When a member wants to leave our system, the system must make sure that this user does not have any borrowed books (all entries in Borrowing file for this user must have its return date filled in)

Administrative actions

9. Overdue books

A list of overdue books -books that were borrowed and their return date has been passed without returning it back- should be available for our system administrators.

10. Most popular books

A list shows the most borrowed 5 books title should be printed at the beginning of our program.

Notes

- It is safer to use fgets instead of gets.
- The program should check for errors and print appropriate messages (e.g. trying to delete an entry that is not in the file, or trying to search for a book that doesn't exist).
- Make sure that book ISBN is unique and user ID is unique.
- Every task of the above tasks should be a separate function.
- You shall use global variable, it will be of a great help for you.
- You should do a menu to enable an easy access for the above functions:
    - Book Management (1)
        - Insert, search (…), add new copies, delete, edit
    - Member Management (2)
        - Register, remove
    - Borrow Management (3)
        - Borrow, return
    - Administrative actions (4)
        - Overdue, popular
    - Save changes (5)
    - Exit (6)
        - Save and exit, exit without saving

Bonus

- Split your work in multiple header files (.h and .c files).
- The program should validate all entered data.
    - Validate a number in the phone number, building, …etc.
    - Validate a sting in the first name, last name, street, city, book title, …etc.
    - Validate the email address (example@domain.com).
    - Validate the date in due date, borrowing date, …etc.
- Implement multi search in task 2, use many fields to search for a book, leaving a field blank means it will not be used in the search.
- Use XML files instead of normal text files.
- Use JSON files instead of normal text files.
- Use excel files instead of normal text files.
- Use git to co-work with your teammate.

Deliverables

- You should work in **groups of two or three**.
- Develop this assignment in C programming language.
- **You should deliver a report** that contains description for your work, sample runs for the program, user manual (how to use the system), and main algorithms used (search algorithm, sort algorithm) using flowchart or pseudocode.
- You should deliver your code online (submission link will be sent later).
- Deadline will be in the week starting from **December 23, 2017** (discussion date will be determined later).
- Start in the project early and come forward with your questions.
- No time will be allowed at the lab during the discussion for any modifications.
- Prior to discussion, you should be prepared to illustrate your work and answer any questions about it. Failing to do so is a strong indication to copying / cheating.
- Late submissions are not accepted.
- All actual programming should be an independent effort. If any kind of cheating is discovered, penalties will apply to the participating students by zero in the project, so delivering a nonworking program is so much better than delivering a copy.

Good luck isA ☺