End of Introduction to Python Assessment
Project – Part implementation of a library book management system

Description

In this project you will implement part of a basic library book management system. The complete system will allow library staff to do the following:

1) Search the library database for books
2) Lend a book to a user (a user can have up to 4 books on loan at any time)
3) Allow a user to return a book
4) Print the status of books within the system to a file

The program is started by giving the following command `python lbms.py -b path_to_books_file -u path_to_users_file`

The books held in the library are stored as comma separated values in the file passed in as the parameter of the -b flag (see Data) and on startup the program reads the csv file to obtain the books stored and details about them. The information contained in this file is referred to as the book database.

The details of the users of the system are stored in a separate csv file also as comma separated values (see Data). The file path is passed to the program using the -u parameter. On startup, the program reads this csv file to obtain the registered users and details about them. The information contained in this file is referred to as the user database.

When the user exits the program, the current states of the book and user databases are stored in their respective files.

Implementation details

Information about the books:
You should create a “Book” dictionary containing keys for the following items (you can include additional fields if your implementation requires them):

1) Author Surname – Two books can have the same author, but their book identification numbers MUST be different
2) Author Firstname
3) Book title – Two books can have the same title, but their book identification numbers (see below) MUST be different
4) Book identification number – this is a unique number that uniquely identifies each book.
5) On loan to – If the book is on loan this field contains the user id of the person it is on loan to

Information about the users:
You should create a “User” dictionary containing at least the following keys (you can include additional fields if your implementation requires them):

1) User surname
2) User firstname
3) User library number
4) Books on loan- there can be up to 4 books on loan at any one time. The book

identification numbers for the books are stored

System requirements
On starting your program, it should display a menu containing the following four items:

1) Search the book database for a book by Author surname
2) Search the user database for a user by surname
3) Lend a book to a user
4) Exit from the system

The requirement of each menu item is elaborated below:

1) Menu option 1 should allow the operator to search the database by the surname of the author (do not worry about case sensitivity – assume all records are in capital letters). For example, a search for STROUSTRUP should return:

```
               Your search for books by STROUSTRUP returned 3 records:
Book ID   Author                     Title                                                     on loan
12        BJARNE STROUSTRUP      THE C++ PROGRAMMING LANGUAGE                                      Yes
15        BJARNE STROUSTRUP      THE C++ PROGRAMMING LANGUAGE                                      No
20        BJARNE STROUSTRUP      PROGRAMMING: PRINCIPLES AND PRACTICE USING C++                   Yes
```

   If the operator enters * then all records in the books database should be printed out.

2) Menu option 2 should allow the operator to search the user database by user surname and return the details of all users matching that surname
   If the operator enters * then all records in the user database should be printed out.

3) Menu option 3 should allow the operator to lend a book to a user. The operator has to specify the book ID and the user ID. If a copy of the book is not on loan and the user has less than 4 books out on loan then the book is lent to the user and the appropriate fields in both the user record and the book record are updated.

4) Menu option 4 exits from the system. On exit, the current state of the book database and the user database should be saved to the appropriate text files as comma separated values.

The data files:
There are two data files containing comma separated values – *project_library_management_system_books_data.csv* and *project_library_management_system_users_data.csv*. On reading the files, the first line should be ignored because it contains headers describing the contents of each column. Each subsequent line contains comma separated values.

What you have to hand in

1) The url to a git repository of your code
2) A report on the project is required. This should contain a description of the project and how you tackled it. You should mention any particular difficulties you encountered and implementation decisions you made to solve them. You should also give an indication of how many hours it took you to do the implementation (i.e. write the source code and check it was working ok) this is required just to gauge the level of difficulty of the project. The report should not be too long, I recommend two pages but use as many or as few pages as you feel you need to use.

You will be required to give an oral presentation (10 minute presentation followed by a discussion) once you have submitted.