Project 1: Book Mobile

Program Description:

For this project, you will be building a program to perform data analytics on a data set of books and book reviews. Users will be able to load both a file containing a series of books, and a file containing a series of reviews for those books. Then, your program should perform the required analyses and return the requested information, such as: listing all books within a literary category; providing detailed information for a specified book; providing the average ratings for each author; and specifying the most helpful reviewer. The user should be able to repeatedly request information until they are satisfied and wish to quit the program.

Program Requirements

- The code for the program should be divided into two files:
    - A **BookMobileFunctions.py** file that contains all of the requested functions except the `main()`
    - A **BookMobile.py** file that contains a `main()` and imports the **BookMobileFunctions** module
- Inside the **BookMobileFunctions.py** file you should have:
    - Import statements for any necessary libraries or modules
    - A function to load a file containing a list of books, that has no parameters, returns a `Dictionary` of `Lists` containing each of the books from the file and a `Set` containing all of the literary categories of the books in the file, and provides the following functionality:
        - Creates a new `Dictionary`
        - Creates a new `Set`
        - Prompts the user for the name of the file containing the book information
            - This file will be in CSV format
            - You are not allowed to hardcode the filenames into the program
        - Check if the file exists using the appropriate functions and if it does not, repeatedly prompt the user for a new filename until the provide the name of a file that does exist
        - Open the file for reading
        - Read one line at a time from the file and for each line:
            - Remove any leading and trailing whitespace from the line
            - Split the line on commas, and store the resulting `List`
            - Using the book's Title as a key, store the List containing the book's information in the `Dictionary`
            - Add the all-lowercase letter literary category to the `Set` of categories
            - Be sure not to include the header line from the file in the `Dictionary` or `Set`
        - Close the file
        - Return the `Dictionary` and `Set`

- A function to load a file containing a list of book reviews, that takes in a `Dictionary` of book information as a parameter, returns a `List` containing each of the book reviews from the file, and provides the following functionality:
    - Creates a new `List`
    - Prompts the user for the name of the file containing the book reviews
        - This file will be in CSV format
        - You are not allowed to hardcode the filenames into the program
    - Check if the file exists using the appropriate functions and if it does not, repeatedly prompt the user for a new filename until the provide the name of a file that does exist
    - Open the file for reading
    - Read one line at a time from the file and for each line:
        - Remove any leading and trailing whitespace from the line
        - Split the line on commas, and store the resulting `List`
        - Check if the Title of the book in the current review is in the `Dictionary` of books, and if it is not, generate a `LookupError` and specify which book is not in the Dictionary in the error's message
        - Otherwise, add the review to the `List` of book reviews
    - Close the file
    - Return the `List` of book reviews
- A function to allow the user to choose a literary category and output all of the books in that category, that takes in a `Dictionary` of book information and a `Set` of literary categories, returns nothing, and provides the following functionality:
    - Informs the user of all of the currently available categories, with a numbered list
    - Prompts the user for the name of which category they wish to see books for
    - Checks if the user entered a category that exists, and if they did not, repeatedly prompt for a category name until they provide one that does exist
    - Output the Title and Author(s), with an appropriate message and in an easy to read format, for each book in the `Dictionary` that is in the user specified literary category
- A function to allow the user to choose a book and output detailed information for that book, that takes in a `Dictionary` of book information and a `List` of book reviews as parameters, returns nothing, and provides the following functionality:
    - Informs the user of all of the currently available book Titles, with a numbered list
    - Prompts the user for the name of which book they wish to see detailed information for
    - Checks if the user entered a book that exists, and if they did not, repeatedly prompt for a book Title until they provide one that does exist
    - Using the `List` of book reviews, calculate the average price and average rating for the user specified book
        - Note not all books have book reviews

        - Output the detailed information for the user specified book, using appropriate messages and in an easy to read format, that includes the book's:
            - Title and Author
            - Brief description
            - Who published the book and when
            - The literary category
            - If at least one review exists:
                - The price, in dollars with two decimals of precision
                - The average rating with one decimal of precision
            - If no reviews exist, inform the user that there are no reviews for the book
    - A function to output the average ratings for each author that has reviews for their books, that takes in a `Dictionary` of book information and a `List` of book reviews as parameters, returns nothing, and provides the following functionality:
        - Using the `Dictionary` of books, determines for each author a list of books that they are the author of
            - It is recommended to use a `Dictionary` to store each author and the list of books they are the author of
            - Note some books have more than one author, and the list of authors is semicolon delimited
        - Using the `List` of book reviews, and the collection of authors and their books, determine the average rating of each author across all of the reviews for all of their books
            - Again it is recommended to use `Dictionaries` to help store this information
        - Output to the user each author and their average, using an appropriate message and in an easy to read format
            - If the author has no reviews for any of their books, inform the user of this using an appropriate message
    - A function to output the name and average helpfulness rating of the most helpful reviewer, that takes in a `List` of book reviews as a parameter, returns nothing, and provides the following functionality:
        - Using the `List` of books reviews, for each reviewer, determine their average helpfulness
            - Average helpfulness should be calculated as the sum of all people who found this reviewer's reviews helpful divided by the sum of all people who reviewed this reviewer's reviews
            - It is recommended to use at least one `Dictionary` to store the sum of all people who found this reviewer's reviews helpful and the sum of all people who reviewed this reviewer's reviews
            - Note some reviews have not been reviewed by other people, and so should not be included in the calculation

        - Output to the user the profile name of the reviewer with the highest helpfulness rating and their average helpfulness as a percentage, with no decimal points of precision, using an appropriate message and in an easy to read format
            - Note, reviewers who have less than 10 people who reviewed their reviews should not be included in the comparison for the highest helpfulness rating
    - A function to output a welcome message, welcoming the user to the book mobile, with an appropriate message, and takes in no parameters and returns nothing
    - A function to output a farewell message, thanking the user for using the book mobile, with an appropriate message, and takes in no parameters and returns nothing
    - A function to output a numbered menu of options and takes in no parameters and returns the number of the option the user chose
        - The list of menu options should be output and should include:
            - Loading the book file
            - Loading the book review file
            - Outputting books by literary category
            - Outputting a book's details
            - Outputting the average author ratings
            - Outputting the most helpful reviewer
            - Quitting
        - The user should be prompted for the number of their option, and then that number returned
- Inside the **BookMobile.py** file you should have:
    - Import statements for any necessary libraries or modules
    - A `main` function which takes in no parameters, returns no values, and provides the following functionality:
        - Contains two variables storing Boolean values, which represent whether the book file and book review file have been loaded
        - Calls the welcome message function
        - Calls the menu function and stores the user's choice
        - In a loop of your choice:
            - In a match/case statement, determine which numeric option the user selected from the menu.
                - If the user chose to read in the book file
                    - Call the appropriate function and store the returned `Dictionary` and `Set`
                    - Update the Boolean variable keeping track of if the book file has been loaded

        - If the user chose to read in the book reviews file:
            - Check if the book file has been loaded, and if not, inform the user
            - Otherwise, try to call the appropriate function and store the returned `List`, if a `LookupError` exception is generated, catch it and output it to the user
            - If the book reviews are successfully read in and stored, update the Boolean variable keeping track of if the book reviews file has been loaded
        - If the user chose to output books in a literary category:
            - Check if the book reviews file has been loaded, and if not, inform the user
            - Otherwise, call the appropriate function
        - If the user chose to output a specific book's details:
            - Check if the book reviews file has been loaded, and if not, inform the user
            - Otherwise, call the appropriate function
        - If the user chose to output all of the average author ratings:
            - Check if the book reviews file has been loaded, and if not, inform the user
            - Otherwise, call the appropriate function
        - If the user chose to output the most helpful reviewer:
            - Check if the book reviews file has been loaded, and if not, inform the user
            - Otherwise, call the appropriate function
        - If the user entered anything else, inform them that they did not choose a valid option
    - Call the menu function and stores the user's new choice
- Calls the farewell message function
- Calls your `main()` function

- Your program should be well documented in terms of comments. For example, good comments in general consist of a header (with your name, date, and brief description), comments for each variable, docstrings for each function, and commented blocks of code.
- You are not allowed to use any globally scoped variables in this project. All variables must be locally scoped.
- You are not allowed to use `pandas` or the `csv` module for this portion of the project
- See sample input and output files to test and calibrate your program

Example Output

See the example output files

Submission

- Your program will be graded largely upon whether it works correctly
- Your program will also be graded based upon your program style. This means that you should use comments (as directed), meaningful variable names, and a consistent indentation style
- You must submit all of your .py files to Canvas by the due date and time, and notebook files will not be accepted
- Projects will be accepted up to two days late, at a 10% penalty per 24-hour period after the due date
- Projects are meant to be individual coding assignments, so no collaboration is allowed. This includes downloading code off of the internet. Any discovered instances of this will be considered cheating and appropriate actions will the taken according to the course syllabus
- You are not allowed to use generative AI on this project. Any discovered instances of this will be considered cheating and appropriate actions will the taken according to the course syllabus
- Be sure that you have tested the version of the program you wish to submit to make sure it works correctly. You will not be allowed to resubmit work after the deadline

Rubric

The entire assignment is worth 100 points and partial credit is possible. No credit will be given for portions of the program that cannot be tested due to the program crashing.

- **Program Executes Successfully**
  - If your program crashes due to syntax errors, you will lose 10 points, but we will attempt to fix minor issues (incorrect indentations, stray character, missing import) so that we can execute and test the program. We will not fix major issues that would require functionality to be further implemented, or a reorganization of logic in your code.
- **Loading the Book File (10 points)**
  - Program prompts the user for the filename and uses a loop to repeatedly reprompt for a new filename if the file does not exist (5 points)
  - Program constructs and returns the `Dictionary` and `Set` based on the information in the user specified file (5 points)
- **Loading the Book Review File (15 points)**
  - Program prompts the user for the filename and uses a loop to repeatedly reprompt for a new filename if the file does not exist (5 points)
  - Program generates an exception if a review exists for a book that is not in the book `Dictionary` (5 points)
  - Program constructs and returns the `List` based on the information in the user specified file (5 points)
- **Menu (5 points)**
  - Program correctly provides a numeric menu to the user, prompts for a choice, and returns the choice (5 points)

- **Outputting the Books in a Category (10 points)**
    - Program presents the user with a numeric list of categories, prompts them for a choice, and repeatedly reprompts if an incorrect choice is made (5 points)
    - Program correctly outputs a list of all book titles and authors that are in the specified category, in an easy to read format (5 points)
- **Outputting a Specific Book's Details (10 points)**
    - Program presents the user with a numeric list of book titles, prompts them for a choice, and repeatedly reprompts if an incorrect choice is made (5 points)
    - Program correctly outputs all of the required information for the user specified book, in an easy to read format (5 points)
- **Outputting Author Average Ratings (10 points)**
    - Program correctly calculates and outputs the name and average ratings for all authors (or no rating if the author has no reviews) in an easy to read format(10 points)
- **Outputting the Most Helpful Reviewer(10 points)**
    - Program correctly calculates and outputs the name and average helpfulness rating for the most helpful reviewer, in an easy to read format(10 points)
- **Introduction and Farewell Messages (5 points)**
    - Program greets and says good bye to the user, user appropriate functions(5 points)
- **Program is Correctly Divided into Two Files (5 points)**
- **Main Functionality (15 points)**
    - Program calls appropriate functions at the appropriate times (5 points)
    - Program handles exceptions and Boolean values correctly (5 points)
    - Program loops repeatedly, and uses a match/case statement to execute the choice the user indicated in the menu, until the user chooses the quit (5 points)
- **Program Contains Sufficient Comments (5 points)**

Bonus

Many of these functions could be improved by using pandas and data frames to process the data. So, for 10 bonus points, write a new, separate version of the program in which:

- Update your function to read in the book file so that it now uses pandas to read the file and stores it in a data frame, rather than a dictionary and set, which is returned by the function
- Update your function to read in the book reviews file so that it now uses pandas to read the file and stores it in a data frame, rather than a list, which is returned by the function
    - Make sure you still perform the requested error checking when reading in reviews
- Update your `main()` function so that it now has variables to store a data frame containing book information and a data frame containing book review information, and stores the returned data frames from the functions to read in the books and book reviews
- Update all of the data analysis functions so that each function takes in the necessary data frame(s) to perform the required analysis
    - Note the majority of the analysis should now be performed using operations on the data frame rather than many dictionaries and lists
- You are not allowed to use any globally scoped variables