Weight: **15%**

Marks: **/100**

Assignment: Functions, Scoping and Abstraction

Type: Group Assignment

Needed Modules: 1, 2 and 3 ONLY

- Students should **ONLY use** programming constructs covered in modules 1, 2 and 3.
- **Submissions using programming concepts that are not covered in modules 1, 2, and 3 will be penalized.**
  - **Penalty with no limit could be applied.**
- **Don’t use while True or while flag particularly if there is a clear loop condition(s)**
  - **Penalty with no limit could be applied**
- **Minimum penalty is 20%**
- **Late submission will not be accepted**

Library Book Management System Scenario

A library has asked you to develop a system for managing books and member checkouts. The program should allow the following features:

- Storing and loading book information into/from a file.
- Adding a new book to the catalog.
- Displaying a list of books in the catalog.
- Editing book details (e.g., title, author, year, availability).
- Searching for a book by its title or ID.
- Deleting a book from the catalog.
- Checking out a book to a member.
- Returning a book.
- Calculating and displaying the total number of books available and checked out.

Equipment and Materials

For this assignment, you will need:

- Python IDE

Instructions

- This assignment will need to be completed outside of class time. See the course schedule and Brightspace for exact due dates.
- There are many details involved in this assignment which could be challenging to get sorted out and working correctly. Therefore, it is strongly recommended to work on the assignment early and in Team.
- Working in a group, review the Scenario and the Program Requirements sections of this document.
- Equally divide the functions between the group members.
  - Each member is responsible for implementing and testing his/her assigned functions and ensuring it is working properly.
  - Members should review the work completed by other members.
  - Integrate all the functions and test the whole program to ensure that the program meets all the requirements.
  - Use constants when it is appropriate.
- The program should exactly implement the provided test runs (check test runs file)
  - Use the values provided in test runs to test your program.
  - Test the program in the same order shown in the test runs file.
- Check your program against the marking criteria for the submission.

Submission

- Only one copy is required per group, and any of the group members may submit the following files to Brightspace:
  - The code of the program that you implemented (.py file)
  - A copy of the output (.txt file)
  - A PDF/DOC peer assessment.

Program Requirements

- The application **MUST use functions** to well structure the program and ensure the code usability.
- The functions (i.e., name, parameters, body) MUST be exactly implemented as specified.

- Book information is loaded from the catalog file (books.txt) when the program starts.
- If the user provides a file that does not exist, the program will run without loading the books from the catalog file (i.e., books.txt)
- The program continues displaying the main menu until the user quits by typing q or Q.
- The user chooses menu options by entering letters (case-insensitive).
- When adding, deleting, or editing a book, the catalog file should be updated.
- The program should prevent duplicate book IDs.
- Searching, editing, and deleting should be done using the book ID.
- Use a two-dimensional list to store book information in memory.
- The program **MUST exactly create the provided sample test runs**
- You must exactly implement the following functions:

Function #	Function	Description
1	**print_menu()**	• Displays the main menu for the Library Book Management System. • Returns the user's selected menu option.
2	**format_record(book info)**	• Formats a book's details as a CSV string. • Receives a list containing book details [title, ID, author, year, availability]. • Returns the book details formatted as a CSV string as follows **title,id,author,year,availability**
3	**list_books(list of books)**	• Displays a list of all books. • Receives a list of current books, where each book is a list of details. The list of books is a 2-dimensional list
4	**add_book(file name, list of books)**	• Adds a new book to the catalog. • Receives the catalog file name to save the books and the current list of books. • It asks the user to enter the book id • If the book exists in the catalog, it returns a string message that indicates the existence of the book.

		• If the book does not exist in the catalog, it asks the user to enter the book details as title, author, year. Then update the list of books and catalog file. Finally, it returns a string message that indicates the successful addition of the book.
5	**edit_book(file name, list of books)**	• Edits the details (title, author, year) of an existing book in the catalog using its ID. • Receives the catalog file name to save the updated books and the current list of books. • Returns either “Book not found” or a message that indicates the successful editing of the book
6	**search_book(list of books)**	• Searches for a book in the catalog by title or ID. • Receives the current list of books. • Returns “Book not Found” if the book does not exist in the catalog • Otherwise, it displays the found book(s) details and return the number of the found books.
7	**delete_book(file name, list of books)**	• Deletes a book from the catalog using its ID. • Receives the catalog file name to save the updated books and the current list of books. • Returns either “Book not found” or a message that indicates the successful deletion of the book.
8	**checkout_book(file name, list of books)**	• Marks a book as checked out using its ID. • Change the availability value to ‘No’. • Receives the catalog file name to save the updated books and the current list of books. • Returns “Book not found”, “Book is already checked out.”, or a message that indicates the successful checking out of the book.
9	**return_book(file name, list of books)**	• Marks a book as returned using its ID. • Change the availability value to ‘Yes’. • Receives the catalog file name to save the updated books and the current list of books. • Returns “Book not found”, “Book is already available.”, or a message that indicates the successful returning of the book.
10	**display_totals(list of books)**	• Displays the number of checked out, available, and total number of books.

		• Receives the current list of books.
11	**load_books(file name)**	• Loads books from a CSV file into a list. • Receives the catalog file name containing book data. • Displays the number of loaded books and returns a list of books, where each book is a list of details. • If the file does not exist, it displays “File does not exist.” And returns an empty list.
12	**update_books(file name, list of books)**	• Saves the current list of books to a file. • Receives the catalog file name to save the books and the current list of books.
13	**main()**	• It is mainly responsible for: • Loading books into a list from the catalog file. • Displaying the application menu until the user decides to exit the application by entering “q” or “Q”. • Calling the appropriate function according to the user selection • Displaying an invalid input message if the user selects an invalid option

- You could create additional functions to reduce the code redundancy.
- The functions which change the books catalog such as add_book(), edit_book(), delete_book(), checkout_book(), return_book() should call update_books() to update the catalog file and also update the list of books
- Ensure appropriate documentation of all functions.

Test Plan

- Check test_runs.pdf for sample/test runs.

Peer assessment

- The peer assessment should be completed accurately and honestly
    - Deduction will be applied if it is not
- Each member should assess the contribution/participation of all other group members.
- Each member should assign a mark out of 10 to other members.
- Accurately describe the task completed by each member
- Please use the following table to complete the peer assessment. Marks/tasks are just samples.

	Member 1 Name	Member 2 Name	Member 3 Name	Completed Tasks
Member 1 Name	N/A	9	10	• Task 1 description • Task 2 description
Member 2 Name	8	N/A	9	• Task 3 description • Task 4 description
Member 3 Name	9	10	N/A	• Task 5 description • Task 6 description
Average	8.5	9.5	9.5	

A student can receive a deduction on their individual grade for the assignment if they do not meet the expected contribution/participation based on the feedback of other group members.

Marking Criteria

- Check Brightspace for marking rubrics.