Mini project ASD

2024-2025

In this mini-project, you are tasked with creating a **Library Management System** in the C programming language. The system should allow basic operations such as adding, removing, and viewing books in a library. The primary objective of this project is to practice and demonstrate your understanding of **data structures** (specifically stacks), **pointers**, and **dynamic memory management** in C.

Features to Implement

You will implement the following functionalities step-by-step:

1. **Add a New Book:** Create a feature that allows the user to add a book to the library. Each book should have a title, an author, and a publication year. Store the book in a dynamic array, simulating a stack structure, where the most recently added book is at the top.

2. **Remove the Last Added Book:** Implement the *Last-In, First-Out (LIFO)* principle to remove the most recently added book from the library. Notify the user about the removal, or indicate if the library is empty.

3. **View All Books:** Provide a way to display all the books currently in the library. Each book should be listed with its title, author, and year of publication. If no books are present, display an appropriate message.

4. **Find a Book by Title:** Add a feature to search for a book in the library by its title. If the book is found, display its details. If not, notify the user that the book does not exist in the library.

5. **View the Most Recently Added Book:** Implement a function to view the details of the most recently added book without removing it from the library. If the library is empty, notify the user.

6. **Clear All Books:** Provide an option to clear all books from the library. This will reset the library to its initial empty state, and you should confirm the action to the user.

Implementation Guidelines

- Use a **structure** to represent a book, with fields for the title, author, and publication year.
- Store books in a **dynamically allocated array** to simulate a stack. Use pointers to manage this array.
- Handle memory allocation and deallocation carefully to avoid memory leaks.
- Use clear and descriptive messages for user interaction to make the program user-friendly.
- Test your program thoroughly to ensure all features work as expected.

Good luck!