****

**REPORT :**

**Library Management System with a combination of stacks and queues for book issuing and returning.**

**Submitted by :**

**Wafi Alam Fathema [C233509]**

Semester: 3rd Section: 3CF

Course Name: Data Structures Lab

Course Code: CSE-2322

Department: CSE

**Submitted to : Asmaul Hosna Sadika**

Adjunct Lecturer

**INTERNATIONAL ISLAMIC UNIVERSITY**

**CHATTAGRAM**

**Introduction:**

The Library Management System is designed to efficiently manage the issuing and returning of books using data structures: stacks and queues. The system allows users to issue books, return them, and view issued and returned books. This project is implemented in C and utilizes a stack to manage issued books and a queue to handle returned books.

**Objectives:**

* To implement an efficient book issuing and returning mechanism.
* To maintain the sequence of book returns using a queue.
* To provide a simple user interface for library operations.

**System Design:** The system employs two key data structures:

* **Stack**: Used to store issued books in a Last-In-First-Out (LIFO) manner.
* **Queue**: Used to store returned books in a First-In-First-Out (FIFO) manner.

**Functionalities**: The system provides the following functionalities:

1. **Issue a Book**: Adds a book to the issued stack.
2. **Return a Book**: Removes a book from the issued stack and adds it to the returned queue.
3. **View Issued Books**: Displays the books currently issued.
4. **View Returned Books**: Displays the books that have been returned.
5. **Exit**: Closes the



**FIGURE:** Library Management System with a combination of stacks and

queues for book issuing and returning

**Features**

* Book Issuing and Returning
* View Issued Books and Returned Books
* User-Friendly Menu
* Efficient Storage

**Input:** The implementation is done in C programming language with the following key components:

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define MAX 5// *Maximum capacity of stack and queue*

// *Structure for a book*

typedef struct {

    int id;

    char title[50];

} Book;

// *Stack for issued books*

typedef struct {

    Book books[MAX];

    int top;

} Stack;

// *Queue for returned books*

typedef struct {

    Book books[MAX];

    int front, rear, count;

} Queue;

// *Function prototypes*

void push(Stack \*s, Book book);

Book pop(Stack \*s);

void enqueue(Queue \*q, Book book);

Book dequeue(Queue \*q);

void displayIssuedBooks(Stack \*s);

void displayReturnedBooks(Queue \*q);

void menu();

int main() {

    Stack issuedBooks = {.top = -1};

    Queue returnedBooks = {.front = 0, .rear = -1, .count = 0};

    int choice;

    do {

        menu();

        printf("Enter your choice: ");

        scanf("%d", &choice);

        Book book;

        switch (choice) {

            case 1:

                if (issuedBooks.top == MAX - 1) {

                    printf("Issue limit reached! Cannot issue more books.\n");

                } else {

                    printf("Enter book ID: ");

                    scanf("%d", &book**.**id);

                    printf("Enter book title: ");

                    scanf(" %[^\n]", book**.**title);// *Read string with spaces*

                    push(&issuedBooks, book);

                    printf("Book issued successfully.\n");

                }

                break;

            case 2:

                if (issuedBooks.top == -1) {

                    printf("No books issued yet!\n");

                } else {

                    book = pop(&issuedBooks);

                    enqueue(&returnedBooks, book);

                    printf("Book '%s' returned successfully.\n", book**.**title);

                }

                break;

            case 3:

                displayIssuedBooks(&issuedBooks);

                break;

            case 4:

                displayReturnedBooks(&returnedBooks);

                break;

            case 5:

                printf("Exiting the system...\n");

                break;

            default:

                printf("Invalid choice! Please try again.\n");

        }

    } while (choice != 5);

    return 0;

}

// *Function to issue a book (Push to Stack)*

void push(Stack \*s, Book book) {

    s->top++;

    s->books[s->top] = book;

}

// *Function to return a book (Pop from Stack)*

Book pop(Stack \*s) {

    return s->books[s->top--];

}

// *Function to add a returned book to Queue*

void enqueue(Queue \*q, Book book) {

    if (q->count == MAX) {

        printf("Return queue is full!\n");

        return;

    }

    q->rear = (q->rear + 1) % MAX;

    q->books[q->rear] = book;

    q->count++;

}

// *Function to remove a returned book from Queue*

Book dequeue(Queue \*q) {

    if (q->count == 0) {

        printf("No returned books available.\n");

        Book empty = {-1, "None"};

        return empty;

    }

    Book book = q->books[q->front];

    q->front = (q->front + 1) % MAX;

    q->count--;

    return book;

}

// *Function to display issued books (Stack)*

void displayIssuedBooks(Stack \*s) {

    if (s->top == -1) {

        printf("No books issued yet.\n");

        return;

    }

    printf("Issued Books (Most recent first):\n");

    for (int i = s->top; i >= 0; i--) {

        printf("ID: %d, Title: %s\n", s->books[i]**.**id, s->books[i]**.**title);

    }

}

// *Function to display returned books (Queue)*

void displayReturnedBooks(Queue \*q) {

    if (q->count == 0) {

        printf("No books returned yet.\n");

        return;

    }

    printf("Returned Books (Oldest first):\n");

    int index = q->front;

    for (int i = 0; i < q->count; i++) {

        printf("ID: %d, Title: %s\n", q->books[index]**.**id, q->books[index]**.**title);

        index = (index + 1) % MAX;

    }

}

// *Function to display menu*

void menu() {

    printf("\n=== Library Management System ===\n");

    printf("1. Issue a Book\n");

    printf("2. Return a Book\n");

    printf("3. View Issued Books\n");

    printf("4. View Returned Books\n");

    printf("5. Exit\n");

}

**Structures Used:**

* Book: Represents a book with an ID and title.
* Stack: Manages issued books.
* Queue: Manages returned books.

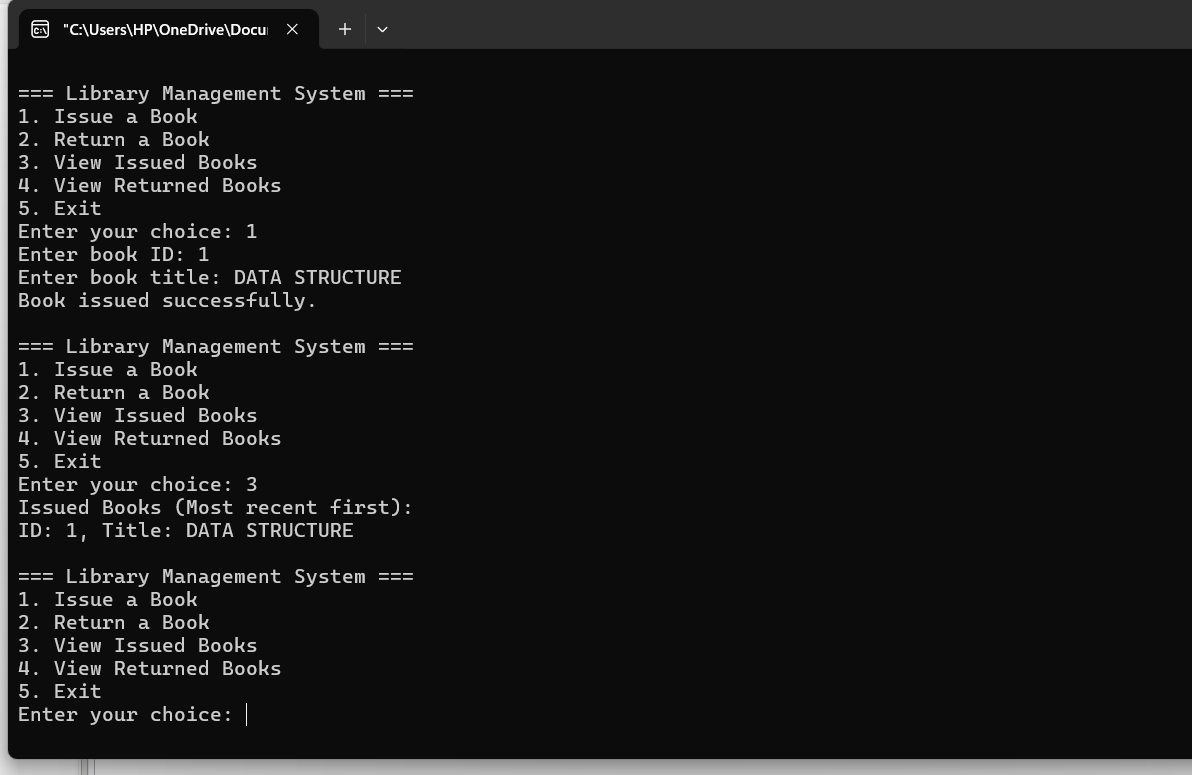
**Key Functions:**

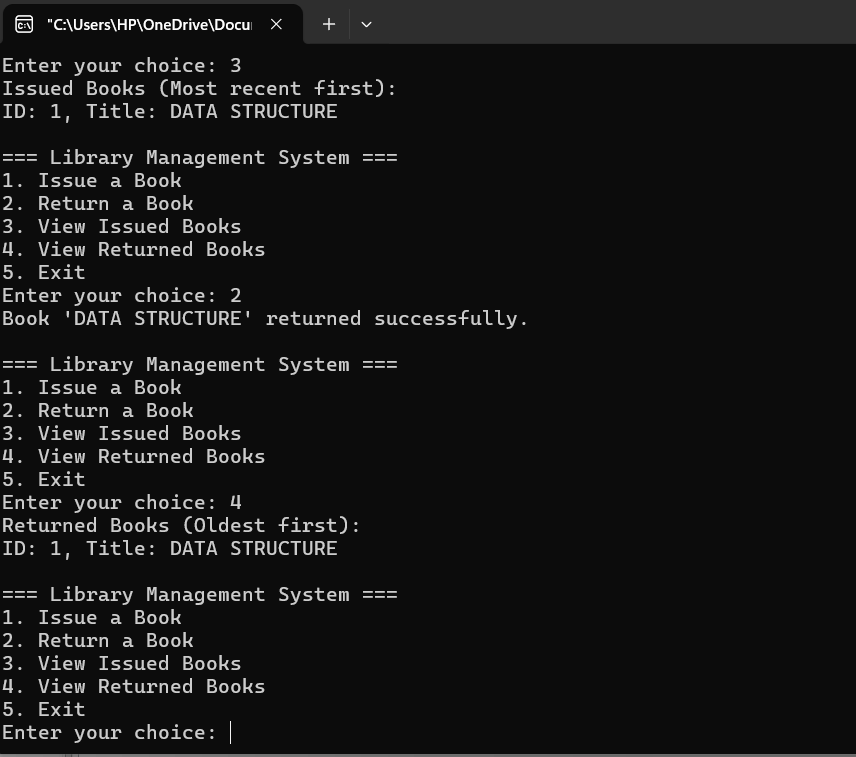
* Push (Stack \*s, Book book): Issues a book by adding it to the stack.
* Pop (Stack \*s): Returns a book by removing it from the stack.
* Enqueue (Queue \*q, Book book): Adds a returned book to the queue.
* Dequeue (Queue \*q): Removes a returned book from the queue.
* DisplayIssuedBooks (Stack \*s): Displays all issued books.
* DisplayReturnedBooks (Queue \*q): Displays all returned books.
* Menu(): Displays the main menu options.

**Code Overview**: The following is an overview of the main logic of the system:

* A **stack** (issuedBooks) is initialized to store issued books.
* A **queue** (returnedBooks) is initialized to store returned books.
* The main() function provides an interactive menu for user actions.
* The system ensures that books are issued up to a maximum capacity and returned in the correct sequence.

**OUTPUT:**

****

****

**Results and Observations:**

* The system successfully maintains book issuing and returning operations.
* The stack ensures that the most recently issued book is the first to be returned.
* The queue maintains the correct order of returned books.
* The system prevents issuing beyond the maximum capacity and returning from an empty stack.

**Conclusion:**

This project demonstrates an efficient Library Management System using stacks and queues. It provides a structured way to manage book transactions, ensuring proper tracking of issued and returned books. Future enhancements may include database integration and user authentication for improved functionality.

**Future Enhancements:**

* Implementing a database for permanent storage.
* Adding a graphical user interface (GUI) for better usability.
* Introducing user authentication to track individual borrowers.
* Expanding the system to handle multiple categories of books.

**THANK YOU SO MUCH**