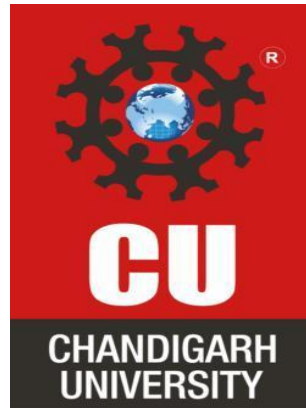


**DEPARTMENT OF
UNIVERSITY INSTITUTE OF COMPUTING
CHANDIGARH UNIVERSITY**



**PROJECT FILE
SUBJECT NAME: Data Structure Lab
SUBJECT CODE: 24CAP-152**

Submitted by:

Name : Jaspreet Singh

UID : 24BCA10517

Section :24BCA-7B

Submitted To:

Name: Mrs Monika Mam

Signature :

Acknowledgement

I would like to express my heartfelt gratitude to **Mrs. Monika Choudhary**, my **Data Structure Lab lecturer**, for her continuous support, valuable guidance, and encouragement during the development of this project titled **“Library Book Tracker System in C.”**

Her insights, motivation, and clear explanations helped me understand the practical implementation of data structures—especially arrays and linked lists—in real-world applications like library management. Her feedback throughout the project greatly enhanced both my understanding and the quality of the final outcome.

I also extend my thanks to **Chandigarh University/ UIC** for providing the necessary resources and a conducive environment to carry out this project

This project has been a great learning experience, and I am truly thankful for her role in helping me successfully complete it.

Name: Jaspreet Singh

UID : 24BCA10516

DATE: 17/04/2025

Abstract

The **Library Book Tracker System** is a mini-project developed using the C programming language to simulate the functioning of a real-world library. The system utilizes fundamental data structure concepts such as arrays and linked lists to manage book records and user transactions effectively.

This project provides an efficient way to **add, remove, search, and display books** in the library. It also handles book issuing and returning operations while updating the number of available and issued copies in real-time. A key feature of the system is the waiting list **mechanism**, which ensures fair allocation of books when all copies are currently issued.

By automating routine library tasks, this project minimizes human errors and improves the overall efficiency of library management. It is an ideal demonstration of how data structures can be applied in solving practical, real-life problems.

PROJECT FILE

Aim: Library Book Tracker System in C.

Introduction

The Library Book Tracker System is a mini project developed in the C programming language using the concepts of data structures. The main purpose of this project is to manage and organize books in a library efficiently. It allows users to add new books, remove books, search for books by title or author, and view all available books in the library.

In addition, the system provides features to issue books to users, return books, and manage a waiting list for books that are currently unavailable. It keeps track of the number of total and issued copies of each book, ensuring accurate and up-to-date information.

This project helped me understand how arrays and structures in C can be used to build practical systems. It also strengthened my logical thinking and programming skills by applying real-world problem-solving techniques.

Objective

The main objective of this project is to design and implement a **Library Book Tracker System** using the C programming language that simplifies the process of managing books in a library.

This system aims to:

- **Maintain records** of all books in the library, including title, author, and available copies.
- **Allow easy searching** of books by title or author.
- **Add and remove books** from the system efficiently.
- **Issue and return books**, while updating the available stock automatically.
- **Handle waiting lists** for books that are currently unavailable.
- **Display complete information** about the books in a user-friendly format.

By achieving these objectives, the project demonstrates the practical use of data structures like arrays and linked lists in solving real-world problems.

Features of the Project

1. Add Book Details

The system allows the librarian to add new books by entering the title, author name, and number of copies available in the library.

2. Remove Book from Library

A book can be removed from the system only if none of its copies are currently issued, ensuring data integrity.

3. Search Book by Title or Author

Enables the user to search for a book using a keyword present in the title or author name, making book retrieval faster and more efficient.

4. Display All Books

Displays all books present in the library along with their details such as title, author, total copies, issued copies, and available copies.

5. Issue Book to User

Books can be issued to users by entering their name and ID. The number of issued copies is updated accordingly. If the book is not available, the user is added to the waiting list.

6. Return Book

When a book is returned, the number of issued copies is decreased. If users are in the waiting list, the next user automatically receives the returned copy.

7. View Waiting List

Displays the list of users who are waiting for a specific book, including their names and user IDs, ensuring fair and organized book issuing.

Real-Life Example: How the Library Book Tracker Works

Imagine you are managing a college library that has multiple students borrowing books every day. Let's say you have the following book:

- **Book Title:** "Let Us C"
 - **Author:** Yashavant Kanetkar
 - **Total Copies in Library:** 2
-

Scenario 1: Normal Issue

- **Student A** comes to borrow "Let Us C"
 - The book is available, so it is issued successfully.
 - Issued Copies: 1 / Available Copies: 1
- **Student B** also wants to borrow "Let Us C"
 - One copy is still available → Book issued
 - Issued Copies: 2 / Available Copies: 0

Scenario 2: Book Not Available – Waiting List Starts

- **Student C** asks for the same book
 - No copies left → Student C is added to the **waiting list**

Scenario 3: Book Return and Auto-Issue to Waitlisted User

- **Student A** returns the book
 - Issued Copies decrease to 1
 - The system automatically issues the book to **Student C** from the waiting list
 - Issued Copies go back to 2

Code

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
```

```
#define MAX_BOOKS 100
#define MAX_WAITING 10
#define MAX_USERS 100
```

```
typedef struct
{
    char name[50];
```

```
    int id;
} User;

typedef struct
{
    char title[100];
    char author[100];
    int totalCopies;
    int issuedCopies;

    User waitingList[MAX_WAITING];
    int waitingCount;
} Book;

Book library[MAX_BOOKS];
int bookCount = 0;

void addBook()
{
    if (bookCount >= MAX_BOOKS)
    {
        printf("\nLibrary is full! Cannot add more books.\n");
        return;
    }
    Book b;
    printf("\nEnter book title: ");
    getchar();
    fgets(b.title, sizeof(b.title), stdin);
    b.title[strcspn(b.title, "\n")] = 0;
    printf("Enter author name: ");
    fgets(b.author, sizeof(b.author), stdin);
    b.author[strcspn(b.author, "\n")] = 0;
    printf("Enter total number of copies: ");
    scanf("%d", &b.totalCopies);
    b.issuedCopies = 0;
    b.waitingCount = 0;
    library[bookCount++] = b;
    printf("Book added successfully!\n");
}

void removeBook()
{
    char title[100];
    getchar();
    printf("\nEnter the title of the book to remove: ");
    fgets(title, sizeof(title), stdin);

    title[strcspn(title, "\n")] = 0;
    for (int i = 0; i < bookCount; i++)
```

```
{
    if (strcmp(library[i].title, title) == 0)
    {
        if (library[i].issuedCopies == 0)
        {
            for (int j = i; j < bookCount - 1; j++)
            {
                library[j] = library[j + 1];
            }
            bookCount--;
            printf("Book removed successfully!\n");
            return;
        }
        else
        {
            printf("Cannot remove book. Some copies are still issued.\n");
            return;
        }
    }
}
printf("Book not found.\n");
}
```

void searchBook()

```
{
    char keyword[100];
    getchar();
    printf("\nEnter title or author keyword to search: ");
    fgets(keyword, sizeof(keyword), stdin);
    keyword[strcspn(keyword, "\n")] = 0;
    printf("\nSearch Results:\n");
    for (int i = 0; i < bookCount; i++)
    {
        if (strstr(library[i].title, keyword) || strstr(library[i].author, keyword))
        {
            printf("Title: %s | Author: %s | Total: %d | Issued: %d | Available: %d\n",
                library[i].title, library[i].author,
                library[i].totalCopies,
                library[i].issuedCopies,
                library[i].totalCopies - library[i].issuedCopies);
        }
    }
}
```

void displayBooks()

```
{
    printf("\nAll Books in Library:\n");
    for (int i = 0; i < bookCount; i++)
    {
```



```
printf("Title: %s | Author: %s | Total: %d | Issued: %d | Available: %d\n",
      library[i].title, library[i].author,
      library[i].totalCopies,
      library[i].issuedCopies,
      library[i].totalCopies - library[i].issuedCopies);
    }
}

void issueBook()
{
    char title[100];
    User u;

    getchar();
    printf("\nEnter book title to issue: ");
    fgets(title, sizeof(title), stdin);
    title[strcspn(title, "\n")] = 0;
    printf("Enter your name: ");
    fgets(u.name, sizeof(u.name), stdin);
    u.name[strcspn(u.name, "\n")] = 0;
    printf("Enter your ID: ");
    scanf("%d", &u.id);

    for (int i = 0; i < bookCount; i++)
    {
        if (strcmp(library[i].title, title) == 0)
        {
            if (library[i].issuedCopies < library[i].totalCopies)
            {
                library[i].issuedCopies++;
                printf("Book issued successfully!\n");
                return;
            }
            else if (library[i].waitingCount < MAX_WAITING)
            {
                library[i].waitingList[library[i].waitingCount++] = u;
                printf("All copies issued. You have been added to the waiting list.\n");
                return;
            }
            else
            {
                printf("Waiting list full. Cannot issue book now.\n");
                return;
            }
        }
    }
    printf("Book not found.\n");
}
```

void returnBook()

```
{
    char title[100];
    getchar();
    printf("\nEnter book title to return: ");

    fgets(title, sizeof(title), stdin);
    title[strcspn(title, "\n")] = 0;

    for (int i = 0; i < bookCount; i++)
    {
        if (strcmp(library[i].title, title) == 0)
        {
            if (library[i].issuedCopies > 0)
            {
                library[i].issuedCopies--;
                if (library[i].waitingCount > 0)
                {
                    User next = library[i].waitingList[0];
                    for (int j = 0; j < library[i].waitingCount - 1; j++)
                    {
                        library[i].waitingList[j] = library[i].waitingList[j + 1];
                    }
                    library[i].waitingCount--;
                    library[i].issuedCopies++;
                    printf("Book automatically issued to: %s (ID: %d) from waiting list.\n",
                        next.name, next.id);
                }
            }
            else
            {
                printf("Book returned successfully!\n");
            }
            return;
        }
        else
        {
            printf("No issued copies to return.\n");
            return;
        }
    }
    printf("Book not found.\n");
}
```

void showWaitingList()

```
{
    char title[100];
    getchar();
```

```
printf("\nEnter book title to view waiting list: ");
fgets(title, sizeof(title), stdin);
title[strcspn(title, "\n")] = 0;

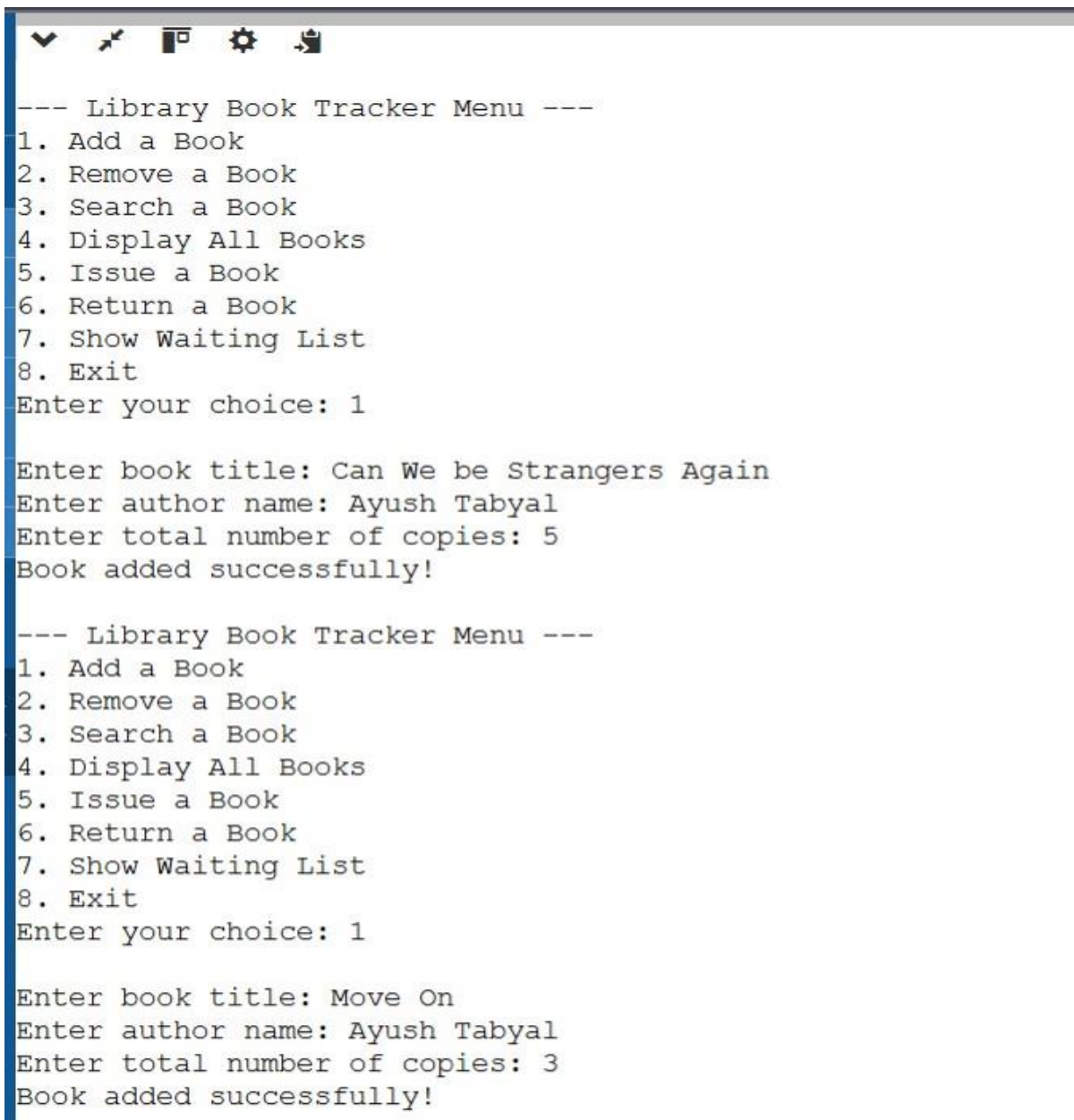
for (int i = 0; i < bookCount; i++)
{
    if (strcmp(library[i].title, title) == 0)
    {
        if (library[i].waitingCount == 0)
        {
            printf("No one is on the waiting list for this book.\n");
        }
        else
        {
            printf("Waiting List for '%s':\n", title);
            for (int j = 0; j < library[i].waitingCount; j++)
            {
                printf("%d. %s (ID: %d)\n", j + 1,
                    library[i].waitingList[j].name,
                    library[i].waitingList[j].id);
            }
        }
        return;
    }
}
printf("Book not found.\n");
}

int main()
{
    int choice;
    do
    {
        printf("\n--- Library Book Tracker Menu ---\n");
        printf("1. Add a Book\n");
        printf("2. Remove a Book\n");
        printf("3. Search a Book\n");
        printf("4. Display All Books\n");
        printf("5. Issue a Book\n");
        printf("6. Return a Book\n");
        printf("7. Show Waiting List\n");
        printf("8. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice)
        {
            case 1: addBook(); break;
            case 2: removeBook(); break;
```

```
        case 3: searchBook(); break;
        case 4: displayBooks(); break;
        case 5: issueBook(); break;
        case 6: returnBook(); break;
        case 7: showWaitingList(); break;
        case 8: printf("Exiting...\n"); break;
        default: printf("Invalid choice. Try again.\n");
    }
}
while (choice != 8);
return 0;
}
```

output



```
--- Library Book Tracker Menu ---
1. Add a Book
2. Remove a Book
3. Search a Book
4. Display All Books
5. Issue a Book
6. Return a Book
7. Show Waiting List
8. Exit
Enter your choice: 1

Enter book title: Can We be Strangers Again
Enter author name: Ayush Tabyal
Enter total number of copies: 5
Book added successfully!

--- Library Book Tracker Menu ---
1. Add a Book
2. Remove a Book
3. Search a Book
4. Display All Books
5. Issue a Book
6. Return a Book
7. Show Waiting List
8. Exit
Enter your choice: 1

Enter book title: Move On
Enter author name: Ayush Tabyal
Enter total number of copies: 3
Book added successfully!
```

```
input
--- Library Book Tracker Menu ---
1. Add a Book
2. Remove a Book
3. Search a Book
4. Display All Books
5. Issue a Book
6. Return a Book
7. Show Waiting List
8. Exit
Enter your choice: 3

Enter title or author keyword to search: Ayush Tabyal

Search Results:
Title: Can We be Strangers Again | Author: Ayush Tabyal | Total: 5 | Issued: 0 | Available: 5
Title: Move On | Author: Ayush Tabyal | Total: 3 | Issued: 0 | Available: 3

--- Library Book Tracker Menu ---
1. Add a Book
2. Remove a Book
3. Search a Book
4. Display All Books
5. Issue a Book
6. Return a Book
7. Show Waiting List
8. Exit
Enter your choice: 5

Enter book title to issue: Can We be Strangers Again
Enter your name: Rohit
Enter your ID: 569
Book issued successfully!
```

```
input
--- Library Book Tracker Menu ---
1. Add a Book
2. Remove a Book
3. Search a Book
4. Display All Books
5. Issue a Book
6. Return a Book
7. Show Waiting List
8. Exit
Enter your choice: 4

All Books in Library:
Title: Can We be Strangers Again | Author: Ayush Tabyal | Total: 5 | Issued: 1 | Available: 4
Title: Move On | Author: Ayush Tabyal | Total: 3 | Issued: 0 | Available: 3

--- Library Book Tracker Menu ---
1. Add a Book
2. Remove a Book
3. Search a Book
4. Display All Books
5. Issue a Book
6. Return a Book
7. Show Waiting List
8. Exit
Enter your choice: 6

Enter book title to return: Can We be Strangers Again
Book returned successfully!
```

```
--- Library Book Tracker Menu ---
1. Add a Book
2. Remove a Book
3. Search a Book
4. Display All Books
5. Issue a Book
6. Return a Book
7. Show Waiting List
8. Exit
Enter your choice: 7

Enter book title to view waiting list: Can ^C

...Program finished with exit code 2
Press ENTER to exit console.
```

Conclusion

The *Library Book Tracker System* effectively demonstrates how core data structures such as arrays and linked lists can be applied in solving real-life problems. Through this project, essential library operations like adding, removing, issuing, and returning books are efficiently managed, along with a waiting list feature for unavailable books.

This system not only automates the routine tasks of a librarian but also minimizes manual errors and improves accuracy. The implementation in C helps in understanding how logic flows in system-level programming and how data can be manipulated using simple structures.

Overall, the project offers a practical learning experience and lays a solid foundation for building more advanced library or inventory management systems in the future.