**NAME: SYED ALI ASAD BUKHARI**

**ROLL.NO: BCS233001**

**COURSE: OOP-SUMMER**

**ASSIGNMENT: 4**

**QUESTION#1**

**TASK-1**

**“LIBRARY MANAGEMENT SYSTEM”**

**PART-A:CODING PART**

**Problem Statement:** Create a C++ class hierarchy to manage a library system. The system should track books, members, and loans.

**Class Structure:**

1. Base Class (Item):

Abstract base class for all library items.

Attributes: item ID, title, availability (Boolean)

Pure virtual functions: getItemDetails(), checkAvailability(),checkout(), checkIn()

2. Derived Classes (Book, Journal):

Inherit from Item.

Additional attributes for books ( author , ISBN) and journals (publisher, issue number).

Implement the pure virtual functions.

3. Class (Member):

Attributes: member ID, name, contact information, loan limit

Member functions: borrowItem(), returnItem(), getLoanHistory()

4. Class (Loan):

Attributes: loan ID, item, member, due date, actual return date

Member function: calculateFine()

**Additional features:**

Reservation system: Allow members to reserve unavailable items.

Category system: Categorize books and journals (e.g. fiction, non-fiction, science).

Search functionality: Implement search by title, author, or ISBN.

Fine calculation: Calculate fines based on overdue days.

Report generation: Generate reports on overdue items, most borrowed items, and member activity.

**Input:**

#include <iostream>

#include <vector>

#include <string>

#include <algorithm>

using namespace std;

struct Book {

string title;

string author;

int totalCopies;

int availableCopies;

};

class Library {

private:

vector<Book> books;

public:

void addBook(const string& title, const string& author, int totalCopies) {

Book newBook;

newBook.title = title;

newBook.author = author;

newBook.totalCopies = totalCopies;

newBook.availableCopies = totalCopies;

books.push\_back(newBook);

cout << "Book added successfully." << endl;

}

void removeBook(const string& title) {

auto it = find\_if(books.begin(), books.end(), [&title](const Book& b) {

return b.title == title;

});

if (it != books.end()) {

books.erase(it);

cout << "Book removed successfully." << endl;

}

else {

cout << "Book not found." << endl;

}

}

void updateBook(const string& title, const string& author, int totalCopies) {

auto it = find\_if(books.begin(), books.end(), [&title](const Book& b) {

return b.title == title;

});

if (it != books.end()) {

it->author = author;

it->totalCopies = totalCopies;

it->availableCopies = totalCopies;

cout << "Book information updated successfully." << endl;

}

else {

cout << "Book not found." << endl;

}

}

void lendBook(const string& title) {

auto it = find\_if(books.begin(), books.end(), [&title](const Book& b) {

return b.title == title;

});

if (it != books.end()) {

if (it->availableCopies > 0) {

it->availableCopies--;

cout << "Book lent successfully." << endl;

}

else {

cout << "No copies available for lending." << endl;

}

}

else {

cout << "Book not found." << endl;

}

}

void returnBook(const string& title) {

auto it = find\_if(books.begin(), books.end(), [&title](const Book& b) {

return b.title == title;

});

if (it != books.end()) {

it->availableCopies++;

cout << "Book returned successfully." << endl;

}

else {

cout << "Book not found." << endl;

}

}

void searchBookAvailability(const string& title) {

auto it = find\_if(books.begin(), books.end(), [&title](const Book& b) {

return b.title == title;

});

if (it != books.end()) {

cout << "Total copies available: " << it->availableCopies << endl;

}

else {

cout << "Book not found." << endl;

}

}

void displayBooks() {

cout << "Books in Library:" << endl;

for (const auto& book : books) {

cout << "Title: " << book.title << ", Author: " << book.author << ", Available Copies: " << book.availableCopies << endl;

}

}

};

int main() {

Library library;

int choice;

string title, author;

int totalCopies;

do {

cout << "\nLibrary Management System\n";

cout << "1. Add Book\n";

cout << "2. Remove Book\n";

cout << "3. Update Book\n";

cout << "4. Lend Book\n";

cout << "5. Return Book\n";

cout << "6. Search Book Availability\n";

cout << "7. Display Books\n";

cout << "8. Exit\n";

cout << "Enter your choice: ";

cin >> choice;

switch (choice) {

case 1:

cout << "Enter title, author, and total copies: ";

cin >> title >> author >> totalCopies;

library.addBook(title, author, totalCopies);

break;

case 2:

cout << "Enter title of the book to remove: ";

cin >> title;

library.removeBook(title);

break;

case 3:

cout << "Enter title of the book to update: ";

cin >> title;

cout << "Enter new title, author, and total copies: ";

cin >> title >> author >> totalCopies;

library.updateBook(title, author, totalCopies);

break;

case 4:

cout << "Enter title of the book to lend: ";

cin >> title;

library.lendBook(title);

break;

case 5:

cout << "Enter title of the book to return: ";

cin >> title;

library.returnBook(title);

break;

case 6:

cout << "Enter title of the book to search: ";

cin >> title;

library.searchBookAvailability(title);

break;

case 7:

library.displayBooks();

break;

case 8:

cout << "Exiting...";

break;

default:

cout << "Invalid choice. Please try again.";

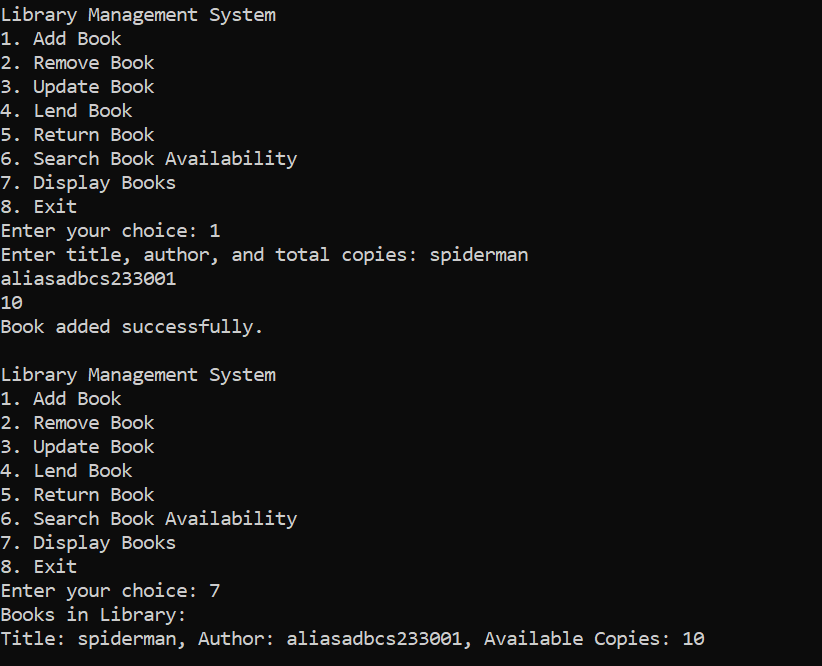
}

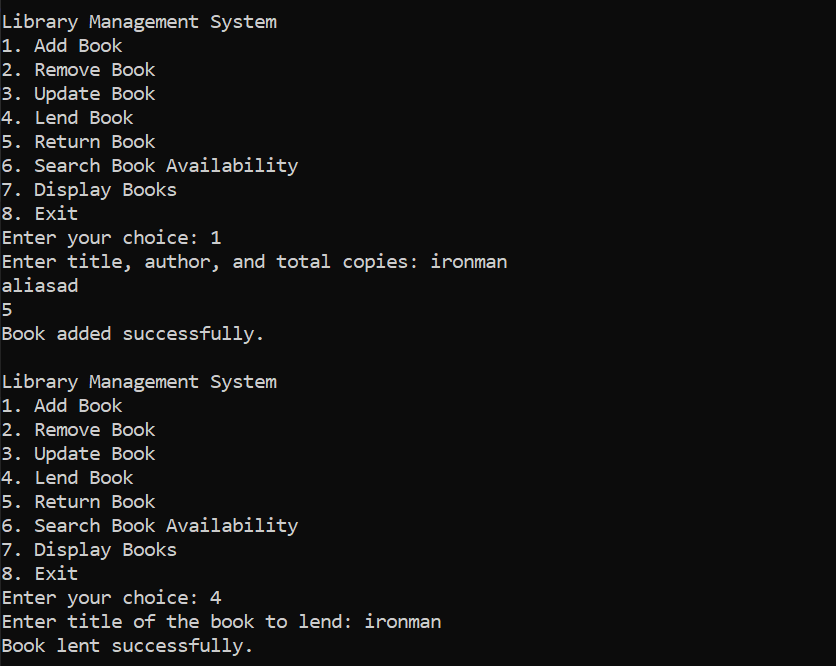
} while (choice != 8);

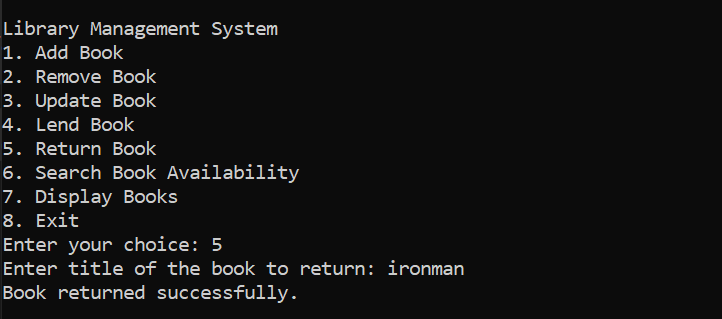
return 0;

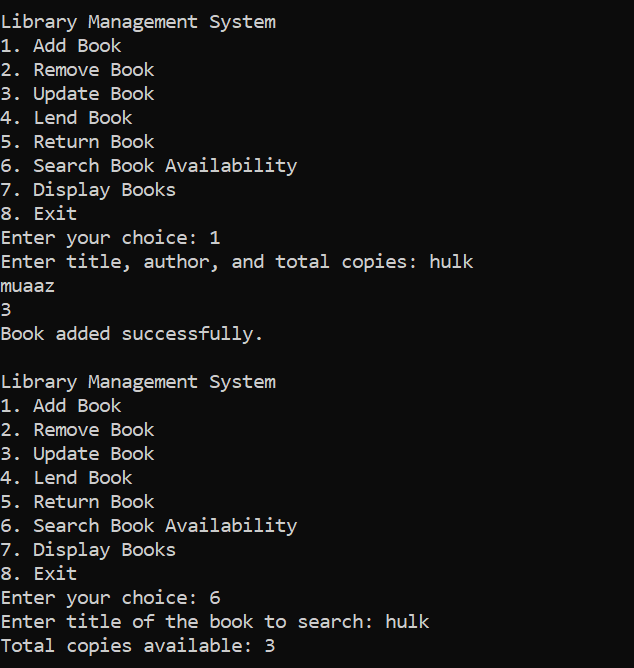
}

**Output:**

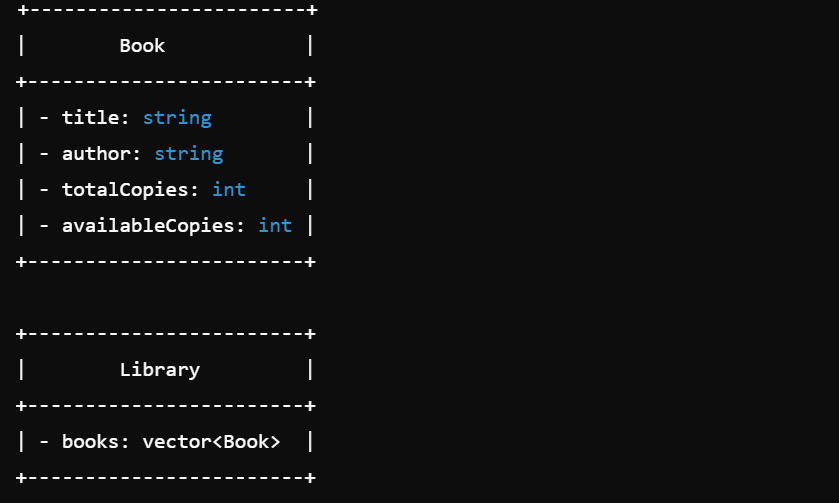


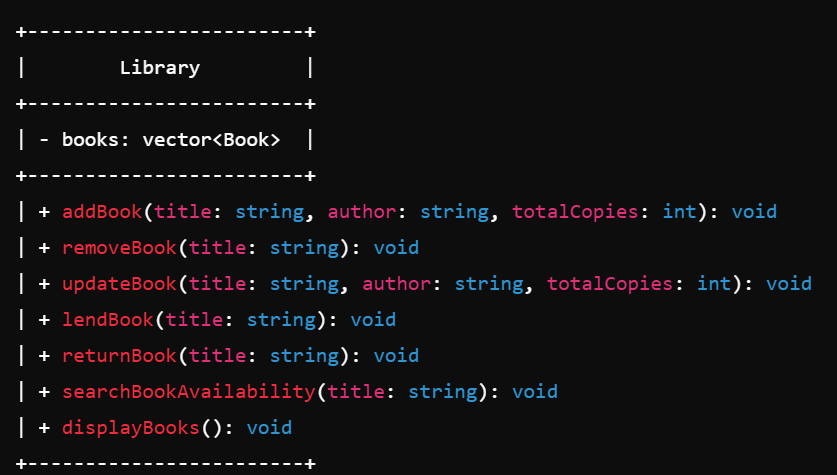






**Class diagram:**





**Sequence diagram:**

