Library Book Rental System (Console-Based Java Application)

Project Title: "Library Book Rental System"

Project Description

A console-based Java application that allows a librarian to manage books and rentals. Users can:

- Add new books to the library
- Search books by title or author
- Rent books to members
- Return books
- Display available books
- Display rented books with member info

The project will cover:

- **Object-Oriented Programming principles**
- **Variables, data types, operators**
- **Control statements**
- **Command line arguments**
- **Static and final variables**
- **Recursive and overloaded methods**
- **Passing objects**
- **Access modifiers**
- **Formatted output**

Syllabus Topic	Implementation
Basic Concepts & Program Structure	`Book` and `Library` classes, `main()` method in `LibraryApp`
Tokens, Statements, Operators	Java syntax, arithmetic for rental calculation
Command Line Arguments	Library name passed from the command line
User Input	`Scanner` for adding/searching books
Escape Sequences	Neat console formatting (`\n`, `\t`)
Comments & Style	Documentation for methods
Data Types	`String`, `int`, `double`, `boolean`
Type Casting	Converting String input to numeric
Variable Scope	Instance, local, static variables
Constants	`final double LATE_FEE = 2.5;`
Formatted Output	`printf()` for book list display
Static Variables/Methods	Static count of total books
Operators	Relational to check availability
Control Statements	Menu via `switch`, loops for continuous use
Classes & Objects	Book, Library, LibraryApp
Constructors	Default and parameterized in `Book`
Nested Classes	Optional inner `Member` class
Final Class/Methods	Final constants
this Keyword	Differentiating instance variables
Method Overloading	Search by title and by author
Recursive Methods	Recursive book search

Syllabus Topic	Implementation
Overriding	`toString()` in `Book`
Passing Objects	Passing `Book` to rental method

Proposed Project Structure

```
LibrarySystem/
|
├── Book.java
├── Library.java
└── LibraryApp.java
```

1. Book.java

- **Fields**: bookId, title, author, price, isAvailable
- **Static Variable**: totalBooks
- **Final Constant**: LATE_FEE
- **Constructors**: default & parameterized
- **Methods**:
  - `displayDetails()`
  - `rentBook()`
  - `returnBook()`
  - Overridden `toString()`

2. Library.java

- **Fields**: `ArrayList<Book> books`
- **Methods**:
  - `addBook(Book b)`
  - `searchBook(String title)` *(overloaded with author)*
  - `rentBook(int bookId)`
  - `returnBook(int bookId)`
  - `displayAvailableBooks()`
  - `displayRentedBooks()`
- **Recursive search** method

3. LibraryApp.java

- **Command Line Argument**: Library name
- **Menu-driven program** using `switch` and `do-while`
- **Scanner** for user input
- **Formatted output** with `printf()`

Sample Console Output

```
===== Welcome to City Library =====
1. Add Book
2. Search Book
3. Rent Book
4. Return Book
5. Show Available Books
6. Show Rented Books
7. Exit
Enter your choice: 1

Enter Book ID: 101
Enter Title: Java Programming
Enter Author: James Gosling
Enter Price: 450
```

Book added successfully!

Evaluation Metrics (100 Marks Total)

Criteria	Marks
Code Functionality – Program runs without errors, all features implemented	30
Syllabus Coverage – Concepts from OOP, data types, operators, control structures, methods, etc. used	20
Code Quality – Indentation, meaningful variable names, comments	10
Project Report Quality – Clear, complete, formatted	15
User Interaction – Menu clarity, input validation, formatted output	10
Innovation/Extra Features – Sorting, file handling, extra search features	10
Presentation/Demonstration – Ability to explain code & logic	5