BEGINNER-LEVEL ROADMAP TO BUILD A SIMPLE BOOKS LIBRARY MANAGEMENT SYSTEM

Project Goal

Create a basic **Books Library Management System** with these features:

1. Add books.
2. Remove books.
3. View all books.
4. Borrow books (assign a book to a person's name).
5. Show borrowed books (display borrowed books with names).
6. Return books (move the book back to the available collection).

Beginner-Friendly Step-by-Step Learning Plan

1. START WITH THE BASICS: CLASSES AND OBJECTS

What to Learn:

- A **class** is a blueprint for objects.
- **Properties** are like variables inside a class to store information.

Practical Task:

- Create a `Book` class with these properties:
  - `Title` (book's name)
  - `Author` (author's name)
  - `IsBorrowed` (true or false to check if it's borrowed)
  - `BorrowedBy` (name of the person who borrowed it)

2. LEARN LISTS TO STORE MULTIPLE BOOKS

- **What to Learn:**
  - A **List** is used to store multiple items (in this case, books).
  - Learn how to add, remove, and loop through items in a list.
- **Practical Task:**
  - Create a list called `availableBooks` to hold all books in the library.
  - Create another list called `borrowedBooks` to store books that are currently borrowed.

3. LEARN METHODS TO PERFORM ACTIONS

What to Learn:

- A **method** is a reusable block of code that performs a specific task (e.g., add, remove, borrow a book).

Practical Task:

- Write these methods in a `Library` class:
  1. `AddBook()` – Add a book to `availableBooks`.
  2. `RemoveBook()` – Remove a book from `availableBooks`.
  3. `ViewAllBooks()` – Loop through `availableBooks` and display the books.

---

4. LEARN CONSOLE INPUT/OUTPUT FOR USER INTERACTION

What to Learn:

- Use `Console.ReadLine()` to take input from the user.
- Use `Console.WriteLine()` to display messages to the user.

Practical Task:

- Allow the user to:
  - Enter the title and author when adding a book.
  - Enter the book's name when borrowing or returning it.

---

5. LEARN CONDITIONAL LOGIC FOR BORROWING AND RETURNING

What to Learn:

- Use `if` statements to check conditions.

Practical Task:

- Borrow a book:
  - Check if the book is in `availableBooks`.
  - If yes, move it to `borrowedBooks` and assign the borrower's name.
  - If no, display a message saying the book is unavailable.
- Return a book:
  - Check if the book is in `borrowedBooks`.

- If yes, move it back to `availableBooks`.

---

6. LEARN LOOPS TO DISPLAY BOOKS

What to Learn:

- Use `foreach` or `for` loops to go through all books in a list.

Practical Task:

- Write a method to display all books in `availableBooks` and `borrowedBooks`.

Plan for Building the System

DAY 1-2: CLASSES AND OBJECTS

- Create a `Book` class with properties: `Title`, `Author`, `IsBorrowed`, `BorrowedBy`.
- Create a `Library` class to manage books.

DAY 3-4: STORING BOOKS IN LISTS

- Create lists for `availableBooks` and `borrowedBooks`.
- Write methods to add and remove books.

DAY 5: BORROWING AND RETURNING BOOKS

- Write methods for `BorrowBook` and `ReturnBook`.
- Use conditionals (`if`) to check if a book is available or already borrowed.

DAY 6: DISPLAYING BOOKS

- Write methods to show:
  - All available books.
  - All borrowed books with the names of the borrowers.

DAY 7: USER INTERACTION

- Use `Console.ReadLine()` to take user input.
- Display menu options (add, remove, borrow, return, view books).

Concepts Summary

1. **Classes and Objects:**
   - Represent books and the library as objects with properties and methods.
2. **Lists:**
   - Store and manage collections of books.
3. **Methods:**
   - Write reusable blocks of code for each feature (add, remove, borrow, return).
4. **Conditional Logic:**
   - Check conditions before performing actions (e.g., "Is the book available?").
5. **Loops:**
   - Use loops to display all books in the library.
6. **Console Input/Output:**
   - Take user input and display results.

Simplified System Features (for Beginners)

1. **Add Book:**
   - Take title and author as input, then add the book to the `availableBooks` list.
2. **Remove Book:**
   - Take the book's name as input, find it in `availableBooks`, and remove it.
3. **Borrow Book:**
   - Take the book's name and borrower's name as input.
   - Check if the book exists in `availableBooks`.
   - Move it to `borrowedBooks` with the borrower's name.
4. **Return Book:**
   - Take the book's name as input.
   - Check if it exists in `borrowedBooks`.
   - Move it back to `availableBooks`.
5. **View All Books:**
   - Show all books in both `availableBooks` and `borrowedBooks`.

Tools You'll Use

- **C# Language Basics:**
  Learn the syntax for classes, lists, methods, loops, and conditionals.
- **Visual Studio Code:**
  Use VS Code to write, debug, and run your program.
- **Console Application:**
  Create a simple console-based interface for the library system.