Project Title: Library

Overview

In this project, you will create a **library system** in Java. The application will enable users to **add books**, **borrow** them, **return** them, and **view all books**—all by interacting with a console-based menu.

Requirements

1. Interface: `Borrowable`

- Methods:
    1. `boolean borrowItem(String borrower)`
    2. `boolean returnItem(String borrower)`
- Any class that represents a borrowable item (`Book`) implements this interface.

2. Custom Exception

- Define a custom checked exception, `BorrowingException`.
    - You might throw this if:
        - A user tries to borrow a book that's already borrowed by someone else.
        - A user tries to return a book that they didn't borrow.

3. `Book` Class (Implements `Borrowable`)

- Fields:
    - `String title`
    - `String author`
    - `String borrowedBy` (the name of the current borrower, if any)
    - `boolean isBorrowed`
- Methods:
    - `borrowItem(String borrower)`:
        - If `isBorrowed == false`, set `isBorrowed = true` and `borrowedBy = borrower`.
        - If `isBorrowed == true`, throw the custom exception or return `false`.
    - `returnItem(String borrower)`:

        - If the book **is** borrowed by that same `borrower`, reset `isBorrowed` to `false` and `borrowedBy` to `null`.
        - Otherwise, throw the exception or return `false`.

4. `Library` Class

- Holds a collection of books, for example a `ArrayList<Book>`.
- Methods:
    1. `addBook(Book book)`: Add a new book to the library's list.
    2. `findBook(String title)`: Return the `Book` object with the matching title (or `null` if not found).
    3. `borrowBook(String title, String borrower)`:
        - Use `findBook(title)` to locate the book.
        - Call `borrowItem(borrower)` on the returned book.
        - Handle or propagate any exception (e.g., `BorrowingException`).
    4. `returnBook(String title, String borrower)`: Similar logic but calls `returnItem(borrower)`.
    5. `listAllBooks()`: Print a summary of each book's title, author, and whether it's borrowed (and by whom).

5. `Main` Class (Driver)

In your `main` method, create a single `Library library = new Library();` instance and drive all interactions through it.

- **Menu** (for example):
    1. **Add Book**
        - Ask the user for `title` and `author`, create a new `Book`, and call `library.addBook(newBook)`.
    2. **Borrow a Book**
        - Ask the user for `title` and `borrowerName`.
        - Call `library.borrowBook(title, borrowerName)`.
        - If a `BorrowingException` or similar custom exception is thrown, **catch** it and display an error message.
    3. **Return a Book**
        - Ask for `title` and `borrowerName`.
        - Call `library.returnBook(title, borrowerName)`.
        - If a exception is thrown, **catch** it and display an error message.
    4. **Show All Books**

        - Call `library.listAllBooks()` and display each book's status (title, author, borrowed by if any).
5. **Exit**
        - End the loop and close the application.

6. Extras (recommended, not hard just more):

- Create a `Magazine` or `DVD` that also implements `Borrowable`, each with any special borrowing rules..
- Track a due date, throw an `OverDueException` if the current date is past due, or if a book is returned late.
- Use simple logs to show more informative messages when operations succeed or fail.