upGrad ENTERPRISE

Assignment Case Study: Library Management System

Problem Statement: Library Management System

Objective: Design and implement a Library Management System using Object-Oriented Programming (OOP) principles in C#. The system should manage different types of library items (e.g., Books and DVDs) and their borrowing rules, demonstrating the use of inheritance, method overriding, abstract classes, and abstract methods.

Requirements:

1. Library Items:

- Create a base class `LibraryItem` to represent common attributes and behaviors of library items, such as:
  - Attributes: `Title` (string), `ItemId` (integer), `IsAvailable` (boolean, default true).
  - A method to display item details (e.g., title, ID, availability status).
- Derive two classes, `Book` and `DVD`, from `LibraryItem`:
  - `Book` should include an additional attribute `Author` (string).
  - `DVD` should include an additional attribute `DurationMinutes` (integer).
  - Each derived class should override the base class method to provide specific details about the item (e.g., include author for books, duration for DVDs).

2. Borrowing Rules:

- Create an abstract class `BorrowingRule` to define the structure for borrowing policies:
  - Attribute: `RuleName` (string).
  - An abstract method `CalculateMaxBorrowingDays()` to return the maximum number of days an item can be borrowed.
  - A non-abstract method to display rule information.
- Derive two classes from `BorrowingRule`:
  - `BookBorrowingRule`: Specifies that books can be borrowed for 14 days.
  - `DVDBorrowingRule`: Specifies that DVDs can be borrowed for 7 days.

3. Functionality:

- Demonstrate inheritance by having `Book` and `DVD` inherit from `LibraryItem`, sharing common attributes and behaviors while adding their own specific properties.

- Demonstrate method overriding by customizing the display of item details in `Book` and `DVD` to include their unique attributes.

- Demonstrate abstract class and abstract methods by using `BorrowingRule` to enforce that derived classes implement the `CalculateMaxBorrowingDays()` method, while providing a common method for rule information.

- Create a main program to:

  - Instantiate `Book` and `DVD` objects and display their details.

  - Simulate changing the availability status of an item and show updated details.

  - Instantiate `BookBorrowingRule` and `DVDBorrowingRule` objects to display borrowing rules and their maximum borrowing days.

Constraints:

- The system should be implemented in C#.

- Ensure the code is simple and clear, suitable for beginners to intermediate learners.

- Use proper OOP principles, ensuring encapsulation (e.g., properties with getters/setters) and clear class hierarchies.

- Avoid hardcoding values in methods; use constructor parameters for initialization.

Expected Output:

- Display details of a book (e.g., "Book - Item ID: 1001, Title: The Great Gatsby, Available: True, Author: F. Scott Fitzgerald").

- Display details of a DVD (e.g., "DVD - Item ID: 2001, Title: Inception, Available: True, Duration: 148 minutes").

- Show updated details after changing availability (e.g., "Book - Item ID: 1001, Title: The Great Gatsby, Available: False, Author: F. Scott Fitzgerald").

- Display borrowing rules (e.g., "Borrowing Rule: Standard Book Rule, Max Borrowing Days: 14" and "Borrowing Rule: Standard DVD Rule, Max Borrowing Days: 7").

upGrad ENTERPRISE

Learning Outcomes:

- Understand how to use inheritance to create a hierarchy of related classes.

- Learn to apply method overriding to customize behavior in derived classes.

- Gain experience with abstract classes and abstract methods to enforce specific implementations in derived classes.

- Develop skills in designing and implementing a simple OOP-based system with real-world applicability.