Course: Object-Oriented Programming **Assignment Type:** Programming Assignments **Due Date:** March 25, 2026 **Total Points:** 100 **Language:** Python or Java (your choice)

Project Overview

Develop a **Library Management System** that demonstrates object-oriented programming principles including encapsulation, inheritance, polymorphism, and abstraction.

Requirements

Part 1: Class Design (30 points)

Design and implement the following classes:

1. Book Class (10 points)

- Attributes: ISBN, title, author, publication year, available status
- Methods:
  - Constructor
  - Getters and setters
  - `display_info()` - prints book details
  - `check_out()` - marks book as unavailable
  - `return_book()` - marks book as available

2. Member Class (10 points)

- Attributes: member ID, name, email, list of borrowed books
- Methods:
  - Constructor
  - `borrow_book(book)` - adds book to borrowed list
  - `return_book(book)` - removes book from borrowed list
  - `display_borrowed_books()` - shows all borrowed books

3. Library Class (10 points)

- Attributes: name, list of books, list of members
- Methods:
  - `add_book(book)` - adds a new book to the library
  - `register_member(member)` - registers a new member
  - `find_book_by_isbn(isbn)` - searches for a book
  - `list_available_books()` - displays all available books
  - `process_checkout(member_id, isbn)` - handles book checkout

– `process_return(member_id, isbn)` - handles book return

Part 2: Inheritance (15 points)

Create specialized book types that inherit from the Book class:

1. TextBook Class

- Additional attributes: subject, edition
- Override `display_info()` to include additional information

2. DigitalBook Class

- Additional attributes: file format, download link
- Override `check_out()` to allow unlimited copies

Part 3: Implementation (35 points)

Implement a main program that demonstrates the following functionality:

1. **Create a library** with at least 5 different books (mix of regular books, textbooks, and digital books) (5 points)
2. **Register at least 3 members** (5 points)
3. **Implement a menu system** with the following options: (15 points)
    - Add a new book
    - Register a new member
    - Check out a book
    - Return a book
    - List all available books
    - Search for a book by ISBN
    - Display member's borrowed books
    - Exit
4. **Error handling** for invalid operations: (10 points)
    - Attempting to check out an unavailable book
    - Invalid member ID or ISBN
    - Member trying to borrow more than 3 books
    - Returning a book that wasn't borrowed

Part 4: Documentation and Testing (20 points)

Documentation (10 points)

- Include docstrings for all classes and methods
- Provide a README.md file explaining:
  - How to run the program
  - Overview of class structure
  - Sample usage scenarios

Testing (10 points)

- Create a separate test file with at least 5 test cases
- Test cases should cover:
  - Normal operations (checkout, return)
  - Edge cases (multiple checkouts, unavailable books)
  - Error conditions

Code Quality Requirements (Points deducted if not met)

- **Naming Conventions:** Use meaningful variable and function names
- **Code Organization:** Proper indentation and structure
- **Comments:** Explain complex logic
- **DRY Principle:** Don't Repeat Yourself - avoid code duplication
- **SOLID Principles:** Demonstrate understanding of OOP principles

---

Submission Guidelines

Deliverables:

1. Source code files (.py or .java)
2. README.md with documentation
3. Test file
4. Brief report (1-2 pages) explaining your design decisions

Submission Format:

- Compress all files into a single ZIP file named: `StudentID_A4_LibrarySystem.zip`
- Submit via the course portal

Grading Rubric

Component	Points	Criteria
Class Design	30	Complete implementation of all required classes with proper encapsulation
Inheritance	15	Proper use of inheritance with meaningful specialized classes
Implementation	35	Functional menu system, error handling, and all required features
Documentation	10	Clear docstrings, comprehensive README, design explanation
Testing	10	Thorough test cases covering normal and edge cases
Total	**100**	

Bonus Opportunity (+10 points)

Implement ONE of the following advanced features: - Persistent storage using files or database - Late fee calculation system for overdue books - Advanced search (by author, title, subject) - GUI interface using tkinter/Swing

---

Academic Integrity

This is an individual assignment. You may discuss concepts with classmates but must write your own code. All submissions will be checked for plagiarism.

Good luck!