COMSATS University, Islamabad

Department of Computer Science

Assignment 3

Deadline: 14/5/2025

[All questions are mapped to CLO 2]

Q1. Design a university course management system that models students, courses, modules, and educational materials. The system must support **polymorphism** to handle different types of students (e.g., UndergraduateStudent, GraduateStudent, ExchangeStudent) and materials (e.g., Textbook, VideoLecture, Assignment), leveraging **inheritance** to define shared behaviors in base classes (Student, Material). Each Course is composed of multiple Module objects, and each Module aggregates a collection of Material objects. Students can enroll in multiple courses, and courses can have many students, managed via an Enrollment class that tracks grades and participation.

Requirements:

1. **Inheritance & Polymorphism**:
    - Create a Student base class with a method calculate_grade() overridden in subclasses (e.g., graduate students might need a thesis defense, exchange students have pass/fail grading).
    - Define a Material base class with a display_content() method, polymorphically implemented by subclasses (e.g., VideoLecture streams video, Assignment shows deadlines).
2. **Composition**:
    - A Course contains 3-5 Module objects. Each Module holds a list of Material objects (e.g., a "Data Structures" course might have modules for "Algorithms" and "Complexity Analysis").
3. **Relationship**:
    - Use an Enrollment class to link Student and Course, storing enrollment dates and grades. Ensure bidirectional navigation (e.g., a student can list all their courses, and a course can list all enrolled students).

Demonstration Tasks:

- Enroll a graduate student and an exchange student in a course with mixed material types.
- Simulate grade calculation for all students in a course, leveraging polymorphism.
- Traverse a course's modules and materials to display content polymorphically.

Deliverables:

- Code implementing the above scenario.
- Test cases proving polymorphism (e.g., calculate_grade() behaves differently per student type) and composition (e.g., deleting a course cascade to its modules).

Q2. A modern library needs a management system to handle its growing collection of materials (books, e-books, audiobooks, DVDs) and diverse patron base (students, faculty, senior citizens, children). The system should automate borrowing, returning, and fine calculations while providing robust search capabilities.

Key Requirements

1. Library Materials

- **Books** (print): Have ISBN, author, publisher, edition
- **E-books**: Include download link and compatible devices
- **Audiobooks**: Contain narrator info and playback length
- **DVDs**: Store director, actors, runtime, and subtitle options

2. Patron Types

- **Students**: Can borrow up to 10 items for 21 days
- **Faculty**: 20-item limit for 42 days
- **Senior Citizens**: 15-item limit, 50% reduced fines
- **Children**: 5-item limit (children's materials only), no fines

3. Core Functionalities

- **Borrowing System**: Track due dates and renewals

- **Fine Calculation**: Automatic computation when items are late
- **Search Engine**: Find materials by various criteria
- **Reporting**: Generate overdue notices and popular item reports

Detailed Use Cases

1. Material Checkout Process

1. Patron presents library card
2. System verifies account standing (no excessive fines)
3. Librarian scans item barcode
4. System:
    - Validates item availability
    - Records loan transaction
    - Sets due date based on patron type
    - Provides printed receipt with due date

2. Return Process

1. Patron returns item to circulation desk
2. Librarian scans item barcode
3. System:
    - Checks for overdue status
    - Calculates fines if applicable
    - Updates item availability
    - If damaged, flags for repair workflow

3. Fine Calculation Rules

- **Books**: 0.25/day(max0.25/*day*(*max*10)
- **DVDs**: 1.00/day(max1.00/*day*(*max*20)
- **E-books**: Auto-returned, no fines
- **Audiobooks**: 0.50/day(max0.50/*day*(*max*15)

- **Faculty/Seniors**: 50% fine reduction
- **Children**: No fines assessed

4. Search Capabilities

- **Basic Search**: Title, author, ISBN
- **Advanced Filters**:
    - Material type
    - Publication year range
    - Language
    - Availability status
    - Genre/category

Special Scenarios

1. Overdue Notifications

- Email reminders at 3 days before due
- First overdue notice at 1 day late
- Account suspension at 14 days overdue
- Collections referral at 30 days overdue

2. Reservation System

- Patrons can place holds on checked-out items
- Automatic notification when available
- 48-hour pickup window before releasing to next patron

3. Interlibrary Loans

- Request materials from partner libraries
- Extended loan periods (6 weeks)
- Special handling fees apply

Technical Requirements

1. Interfaces to Implement

1. **Loanable**
    - checkOut()
    - checkIn()
    - isOverdue()
    - renew()
2. **FineCalculatable**
    - calculateFine()
    - getFineRate()
    - getMaxFine()
3. **Searchable**
    - searchByTitle()
    - searchByAuthor()
    - filterByAvailability()
4. **Reportable**
    - generateOverdueReport()
    - generatePopularItemsReport()
    - generateUsageStatistics()

2. Database Requirements

- Materials catalog
- Patron records
- Transaction history
- Reservation queue
- Fine payment records

Example Workflow

1. **Faculty Member Checks Out DVD**
    - Professor Smith (faculty) borrows documentary DVD

- System sets 42-day loan period
- Automatic renewal available if no holds exist

2. **Student Returns Overdue Book**
    - Book is 5 days late
    - System calculates 1.25fine(5×1.25*fine*(5×0.25)
    - Places hold on student account until paid
3. **Child Borrows Picture Book**
    - 7-year-old checks out children's book
    - 14-day loan period
    - No fines accrue if returned late
4. **Librarian Runs Monthly Report**
    - Identifies 23 overdue items
    - Flags 5 long-overdue for replacement
    - Notes most popular genre is mystery