Assignment-2

CSE201 Advanced Programming

Release date: 22 Sep
Due by: 2 Oct,11:59 P.M.

This assignment is a take-home lab assignment. No extensions whatsoever will be provided. Any submission after the deadline will not be evaluated. If there is any ambiguity or inconsistency in a question, please seek clarification from the teaching staff. Please read the entire text below very carefully before starting its implementation.

Plagiarism: All submitted lab assignments are expected to result from your effort. You should never misrepresent someone else's work as your own. In case any plagiarism case is detected, it will be dealt with as per IIITD plagiarism policy and without any relaxations:

Academic Dishonesty in Courses Homework / Project-report / Submissions
Please note that you cannot discuss the lab assignment's design/solution (e.g., classroom page discussions). Anyone who is found doing this will be treated as a plagiarism case.
AI plagiarism will also be checked strictly.No Excuses!

NOTE that we will ONLY respond to valid questions. Make sure you ask all your doubts in advance. **Doubts on the day of submission will not be entertained**.
With doubts coming, some parts of the assignment may get updated. So, make sure you regularly follow up on doubts in the classroom. If the assignment is updated in between, it will be announced.

Problem Statement

You are tasked with enhancing the university course registration system you implemented in Assignment 1 by introducing new features. These enhancements will require the use of generic programming, object classes, and exception handling. You are not expected to change the existing functionalities but to build upon them, adding these advanced concepts.

Your new additions must integrate smoothly with the current system and ensure proper usage of the object-oriented programming principles covered in class.

PART 1 – Enhancing the Program (50 marks)

General Requirements:

You will enhance the system by implementing the following new features:

1. **Generic Feedback System (20 marks)**

- **Concept**: **Generic Programming**
- **Requirements**:
    - Add a feature where **students can give feedback** on courses they have completed. Use a **generic class** to manage this feedback, allowing it to store different types of data (e.g., numeric ratings, textual comments).
    - Each student can give a numeric rating (e.g., 1–5) and/or textual feedback (e.g., "Great course!").
    - Professors should be able to view this feedback through their user interface.

2. **Enhanced User Role Management with Object Class (20 marks)**

- **Concept**: **Object Class, Inheritance**
- **Requirements**:
    - Without changing the core `Student`, `Professor`, and `Administrator` classes, introduce a **new type of user**: a **Teaching Assistant (TA)**.
    - The TA will be able to assist professors by viewing and managing student grades but will not have the full privileges of a professor (e.g., they cannot update course details).
    - Implement the TA role using **inheritance** by extending the `Student` class. TAs should have all the capabilities of students, with additional functionalities.

- Ensure that the **TA class inherits from the Student class** and adds extra methods to handle grading assistance.

3. **Robust Exception Handling for Course Registration and Login (20 marks)**

- **Concept**: **Exception Handling**
- **Requirements**:
  - Add exception handling for the following scenarios:
    1. **Invalid course registration**: If a student tries to register for a course that is already full, throws an exception (e.g., `CourseFullException`).
    2. **Invalid login**: If a user tries to log in with incorrect credentials, throw a custom exception (e.g., `InvalidLoginException`).
    3. **Course drop failures**: If a student tries to drop a course after a certain deadline, throw an exception (e.g., `DropDeadlinePassedException`).
  - Create **custom exception classes** for each scenario and handle them using **try-catch blocks** in your program.

Submission Guidelines:

1. **Code**: Submit your enhanced code in a `.zip` file. Ensure that the directory structure is clean, and all new features are documented properly.
2. **README**: Include a README file explaining how you have incorporated **generic programming**, **object classes**, and **exception handling** into your solution.
3. **Demonstration**: Provide examples demonstrating how your new enhancements work (e.g., feedback system, TA functionality, exception handling).

The solution to complex problems is simple, mostly simple.

ALL THE BEST!!