University Management System

Objective: Develop a University Management System in Java that demonstrates the four main principles of Object-Oriented Programming (OOP): Inheritance, Abstraction, Polymorphism, and Encapsulation.

Requirements:

Classes and Inheritance:

Create an abstract class User that represents a generic user in the university. This class should include the following attributes:

id (int): Unique identifier for each user.

name (String): Name of the user.

email (String): Email address of the user.

The User class should have:

An abstract method getDetails() to display user details.

Getters and setters for all attributes to demonstrate encapsulation.

Create three subclasses that inherit from User:

Student: Represents a student. It should have additional attributes:

studentId (String): Unique identifier for the student.

courses (List<String>): A list of courses the student is enrolled in.

Professor: Represents a professor. It should have additional attributes:

professorId (String): Unique identifier for the professor.

department (String): The department to which the professor belongs.

coursesTaught (List<String>): A list of courses the professor teaches.

Staff: Represents a staff member. It should have additional attributes:

staffId (String): Unique identifier for the staff member.

role (String): The role of the staff member (e.g., "Admin", "Clerk").

Interface and Abstraction:

Define an interface Course with the following methods:

void getCourseDetails(): Displays course details.

void assignProfessor(Professor professor): Assigns a professor to the course.

void enrollStudent(Student student): Enrolls a student in the course.

Create a class ProgrammingCourse that implements the Course interface and provides the course details (e.g., course name, course code).

Polymorphism:

Implement method overriding for the getDetails() method in the Student, Professor, and Staff classes to display detailed information specific to each type of user.

Demonstrate polymorphism by creating a method in a class (e.g., Department) that takes a parameter of type User and calls the getDetails() method.

Encapsulation:

Use encapsulation to protect the data of each class. Make all fields private and provide public getter and setter methods to access and modify the data.

Additional Classes:

Create a class Department that represents a university department and includes:

name (String): The department name.

courses (List<Course>): A list of courses offered by the department.

users (List<User>): A list of users (students, professors, staff) associated with the department.

Methods to add courses, add users, and display department details.

Create a class University that contains:

name (String): The university name.

departments (List<Department>): A list of departments in the university.

Methods to add departments and display university details.

Main Method Implementation:

In the Main class, demonstrate the following functionalities:

Create instances of Student, Professor, and Staff.

Create instances of ProgrammingCourse and demonstrate assigning professors and enrolling students.

Add users and courses to a department, and add the department to a university.

Display all information using the overridden getDetails() method and other methods.

you can replace any List<User> to an array