OOPs with Python - Assignment

Q1: Design a Python program for a Hotel Management System that consists of a *Hotel* class and a *Room* class. The *Hotel* class should include methods for adding rooms, removing rooms, displaying rooms, and categorizing rooms as vacant or occupied. The *Room* class should include methods for check-ins, check-outs, and determining room status. Implement the program to allow for managing rooms in the hotel, including their capacity, occupancy, and availability. The system should accommodate various room operations, check-ins, check-outs, and provide information about each room's status.

Create instances of the *Hotel* class, add rooms, and perform check-ins and check-outs in the Python program. Additionally, demonstrate the ability to display room details and categorize rooms as vacant or occupied.

Q2: Design a Python program for an Airport Management System that includes an *Airport* class and a *Flight* class. The system should apply encapsulation by using private and public attributes and methods appropriately. The *Airport* class should contain Airport name, list of flights as private attributes, a private method to find a flight by its flight number, and public methods to add and remove flights, list upcoming and completed flights, and display all flight details. The *Flight* class should contain Flight number, destination, departure time, and arrival time as private attributes, a private method to determine if a flight is upcoming, and public methods to update the departure time, arrival time, and to display the details of the flight, including flight number, destination, and times.

Implement the classes to manage flight operations, including adding flights, updating schedules, and displaying categorized flight information. Create instances of the *Airport* class, perform flight operations, and demonstrate data encapsulation through private and public members.

Q3: Design a Python program for a problem that deals with geometric shapes and their calculations and demonstarates inheritance and polymorphism. First, create an abstract base class *Shape* with a public abstract method *area()*. Next, create derived classes for the following geometric shapes, each implementing the *area()* method:

- *Circle* with attributes radius and pi (for the value of π).
- *Rectangle* with attributes length and width.
- *Triangle* with attributes base and height.

Then, create a class *ShapeAreaCalculator* with a public method *calculate_area()* which accepts a shape object as a parameter and calculates its area using polymorphism.

Implement a menu-driven program that allows the user to choose a shape (Circle, Rectangle, or Triangle) and enter the required dimensions (e.g., radius, length, width, height). Finally, use the *ShapeAreaCalculator* class to calculate and display the area of the selected shape. Demonstrate inheritance, abstraction, and polymorphism by creating instances of various shapes and computing their areas dynamically.

XXXXXXXXXXXX