Final Practice questions - Object-oriented Programming - Python

Do all if you can

Question 1 (20 Marks)

a) Construct the following classes: i. **Person** class with attributes name, age, and gender. [2 Marks each]
ii. **Employee** class with attributes employee_id and department (inheriting from Person). [2 Marks each]
b) Write a method assign_task(task) in Employee class to assign a task to an employee. [5 Marks]
c) Create two employee objects, assign them tasks, and display their information. [5 Marks]
d) Write a method display_employee_info() that shows the employee's name, department, and assigned tasks. [3 Marks]
e) Handle edge cases such as assigning the same task twice. [3 Marks]

Question 2 (20 Marks)

a) Construct the following: i. **Vehicle** class with attributes make, model, and year. [2 Marks each]
ii. **Car** class that inherits from Vehicle and includes an additional attribute fuel_type. [2 Marks each]
b) Write a method refuel(amount) in Car class to add fuel. [5 Marks]
c) Create a car object and simulate driving it and refueling it. [5 Marks]
d) Write a method display_vehicle_info() to show details of the car, including fuel level. [3 Marks]
e) Handle cases where fuel level exceeds the tank capacity. [3 Marks]

Question 3 (20 Marks)

a) Construct the following classes: i. **ShopItem** with attributes name, price, and stock_quantity. [2 Marks each]
ii. **Electronics** class that inherits from ShopItem and adds a warranty period attribute. [2 Marks each]
b) Write a method sell_item(quantity) in Electronics class that reduces stock and checks if the item is out of stock. [5 Marks]
c) Create two Electronics objects and sell some quantities, showing how stock changes. [5 Marks]
d) Write a method display_item_info() to show item details and current stock. [3 Marks]
e) Handle cases where stock quantity becomes negative. [3 Marks]

Question 4 (20 Marks)

a) Construct the following classes: i. **Author** class with attributes name and nationality. [2 Marks each]
ii. **Book** class with attributes title, genre, and availability (inheriting from Author). [2 Marks each]
b) Write a method borrow_book() in Book class that changes the availability status. [5 Marks]
c) Create two Book objects and simulate borrowing one of them. [5 Marks]
d) Write a method display_book_info() that displays the book's title, author, and availability. [3 Marks]
e) Handle cases where the book is already borrowed. [3 Marks]

Question 5 (20 Marks)

a) Construct the following classes: i. **Animal** class with attributes species, name, and habitat. [2 Marks each]
ii. **Mammal** class that inherits from Animal and includes a fur_color attribute. [2 Marks each]
b) Write a method feed() in Mammal class that changes the animal's state to "fed". [5 Marks]
c) Create two Mammal objects and feed them. [5 Marks]
d) Write a method display_animal_info() that shows the species, name, and current state (fed or not). [3 Marks]
e) Handle cases where an already fed animal is fed again. [3 Marks]

Question 6 (20 Marks)

a) Construct the following classes: i. **BankAccount** class with attributes account_number, balance, and account_holder. [2 Marks each]
ii. **SavingsAccount** class that inherits from BankAccount and adds an interest_rate attribute. [2 Marks each]
b) Write a method deposit(amount) and withdraw(amount) in SavingsAccount. [5 Marks]
c) Create two SavingsAccount objects and perform transactions (deposit and withdraw). [5 Marks]
d) Write a method display_account_info() that shows the account details and current balance. [3 Marks]
e) Handle cases where the withdrawal amount exceeds the balance. [3 Marks]

Question 7 (20 Marks)

a) Construct the following classes: i. **Publisher** class with attributes name, location, and founded_year. [2 Marks each]
ii. **Magazine** class that inherits from Publisher and includes an issue_number attribute. [2 Marks each]
b) Write a method publish() in Magazine class to simulate publishing a new issue. [5 Marks]
c) Create two Magazine objects and simulate publishing for each. [5 Marks]

d) Write a method display_magazine_info() to show magazine and publisher details. [3 Marks]
e) Handle cases where a magazine tries to publish without an issue number. [3 Marks]

Question 8 (20 Marks)

a) Construct the following classes: i. **AccountHolder** class with attributes name, age, and address. [2 Marks each]
ii. **LoanAccount** class that inherits from AccountHolder and includes loan_amount and interest_rate attributes. [2 Marks each]
b) Write methods apply_for_loan(amount) and repay_loan(amount) in LoanAccount. [5 Marks]
c) Create two LoanAccount objects, apply for loans, and repay some amount. [5 Marks]
d) Write a method display_loan_info() that shows loan details, including balance after repayment. [3 Marks]
e) Handle cases where repayment amount exceeds the loan balance. [3 Marks]

Question 9 (20 Marks)

a) Construct the following classes: i. **TeamMember** class with attributes name, role, and skill_level. [2 Marks each]
ii. **Project** class with attributes project_name and project_deadline. [2 Marks each]
b) Write a method assign_member(team_member) in Project class to assign members to the project. [5 Marks]
c) Create a project and assign team members to it. [5 Marks]
d) Write a method display_project_info() that shows the project name, deadline, and assigned members. [3 Marks]
e) Handle cases where a member is assigned to the project more than once. [3 Marks]

Question 10 (20 Marks)

a) Construct the following classes: i. **Passenger** class with attributes name, ticket_number, and seat_number. [2 Marks each]
ii. **Flight** class with attributes flight_number and destination. [2 Marks each]
b) Write a method assign_seat(passenger) in Flight class to assign passengers to seats. [5 Marks]
c) Create a flight and assign seats to two passengers. [5 Marks]
d) Write a method display_flight_info() to show flight details and assigned passengers. [3 Marks]
e) Handle cases where a seat is already taken. [3 Marks]

These questions challenge students to apply OOP concepts in Python, such as class construction, method implementation, inheritance, and handling edge cases.

Question 11 (20 Marks)

a) Construct the following classes:
i. **Customer** class with attributes name, email, and phone_number. [2 Marks each]
ii. **Order** class with attributes order_number, order_date, and status. [2 Marks each]
b) Write methods place_order() and cancel_order() in the Order class. [5 Marks]
c) Create two customers and have each place and cancel orders. [5 Marks]
d) Write a method display_order_details() that shows order information for a customer. [3 Marks]
e) Handle cases where a customer tries to cancel an order that hasn't been placed. [3 Marks]

Question 12 (20 Marks)

a) Construct the following classes:
i. **Teacher** class with attributes name, subject, and years_of_experience. [2 Marks each]
ii. **Classroom** class with attributes class_code and student_list. [2 Marks each]
b) Write a method add_student(student_name) and remove_student(student_name) in Classroom class. [5 Marks]
c) Create a classroom and add and remove students from it. [5 Marks]
d) Write a method display_class_info() to show the classroom details and students. [3 Marks]
e) Handle cases where a student is removed who isn't in the class. [3 Marks]

Question 13 (20 Marks)

a) Construct the following classes:
i. **Doctor** class with attributes name, specialization, and years_of_experience. [2 Marks each]
ii. **Appointment** class with attributes patient_name, date, and status. [2 Marks each]
b) Write a method schedule_appointment() and cancel_appointment() in Appointment class. [5 Marks]
c) Create a doctor and schedule two appointments for them. Cancel one appointment. [5 Marks]
d) Write a method display_appointments() to show the doctor's scheduled appointments. [3 Marks]
e) Handle cases where an appointment is canceled that was not scheduled. [3 Marks]

Question 14 (20 Marks)

a) Construct the following classes:
i. **Player** class with attributes name, team, and position. [2 Marks each]
ii. **Game** class with attributes game_date, location, and teams. [2 Marks each]
b) Write a method add_team(team_name) and record_score(team_name, score) in the Game class. [5 Marks]
c) Create a game and record scores for two teams. [5 Marks]

d) Write a method display_game_info() to show game details and final score. [3 Marks]
e) Handle cases where a score is recorded for a team not in the game. [3 Marks]

Question 15 (20 Marks)

a) Construct the following classes:
i. **Customer** class with attributes name, email, and balance. [2 Marks each]
ii. **EcommerceSite** class with attributes site_name and available_items. [2 Marks each]
b) Write methods purchase_item(item, quantity) and refund_item(item, quantity) in EcommerceSite. [5 Marks]
c) Create two customers and allow them to purchase and refund items from the site. [5 Marks]
d) Write a method display_purchase_history() to show a customer's purchase history. [3 Marks]
e) Handle cases where a customer attempts to purchase more than available stock. [3 Marks]

Question 16 (20 Marks)

a) Construct the following classes:
i. **Athlete** class with attributes name, sport, and ranking. [2 Marks each]
ii. **Competition** class with attributes competition_name, date, and athletes. [2 Marks each]
b) Write a method register_athlete() and remove_athlete() in Competition class. [5 Marks]
c) Create a competition and register and remove athletes from it. [5 Marks]
d) Write a method display_competition_details() to show the competition's details and participating athletes. [3 Marks]
e) Handle cases where an athlete is removed but isn't registered in the competition. [3 Marks]

Question 17 (20 Marks)

a) Construct the following classes:
i. **Company** class with attributes company_name, industry, and total_employees. [2 Marks each]
ii. **Department** class with attributes department_name, manager_name, and employee_count. [2 Marks each]
b) Write methods add_employee(employee_name) and remove_employee(employee_name) in Department class. [5 Marks]
c) Create a company and add and remove employees from a department. [5 Marks]
d) Write a method display_department_info() to show department details and list of employees. [3 Marks]
e) Handle cases where an employee is removed but not part of the department. [3 Marks]

Question 18 (20 Marks)

a) Construct the following classes:
i. **Product** class with attributes product_name, product_code, and price. [2 Marks each]
ii. **Inventory** class with attributes inventory_id and product_list. [2 Marks each]
b) Write methods add_product(product_name, quantity) and remove_product(product_name, quantity) in Inventory class. [5 Marks]
c) Create an inventory and add and remove products from it. [5 Marks]
d) Write a method display_inventory() to show the products and quantities in stock. [3 Marks]
e) Handle cases where a product is removed but isn't in the inventory. [3 Marks]

Question 19 (20 Marks)

a) Construct the following classes:
i. **Hotel** class with attributes hotel_name, location, and rooms_available. [2 Marks each]
ii. **Reservation** class with attributes guest_name, check_in_date, and room_number. [2 Marks each]
b) Write methods book_room() and cancel_reservation() in Reservation class. [5 Marks]
c) Create a hotel and book and cancel reservations for two guests. [5 Marks]
d) Write a method display_reservation_details() to show reservation details for a guest. [3 Marks]
e) Handle cases where a reservation is canceled that was never booked. [3 Marks]

Question 20 (20 Marks)

a) Construct the following classes:
i. **Customer** class with attributes name, email, and phone. [2 Marks each]
ii. **Account** class with attributes account_number, balance, and account_type. [2 Marks each]
b) Write methods deposit() and withdraw() in Account class. [5 Marks]
c) Create two customers and perform deposit and withdrawal operations for each. [5 Marks]
d) Write a method display_account_balance() to show account details and current balance. [3 Marks]
e) Handle cases where withdrawal exceeds the balance. [3 Marks]

Question 21 (20 Marks)

a) Construct the following classes: i. **Author** class with attributes name, nationality, and birth_year. [2 Marks each]
ii. **Article** class that inherits from Author and includes attributes title and published_year. [2 Marks each]

b) Write a method publish_article(title, year) in Article class to simulate publishing. [5 Marks]
c) Create two article objects and simulate publishing for both. [5 Marks]
d) Write a method display_article_info() to show the article's title, author, and publication year. [3 Marks]
e) Handle cases where the publication year is in the future. [3 Marks]

Question 22 (20 Marks)

a) Construct the following classes: i. **User** class with attributes username, email, and password. [2 Marks each]
ii. **Admin** class that inherits from User and adds the attribute role (set to 'Admin' by default). [2 Marks each]
b) Write methods create_user(username, email, password) and delete_user(username) in Admin class. [5 Marks]
c) Create an admin object and use it to create and delete user accounts. [5 Marks]
d) Write a method display_user_info() that shows user details. [3 Marks]
e) Handle cases where a user account to be deleted does not exist. [3 Marks]

Question 23 (20 Marks)

a) Construct the following classes: i. **Device** class with attributes device_name, model, and status (on/off). [2 Marks each]
ii. **Smartphone** class that inherits from Device and adds attributes battery_level and storage_capacity. [2 Marks each]
b) Write a method turn_on() in Smartphone class to change the device's status to "on". [5 Marks]
c) Create a smartphone object and simulate turning it on and off. [5 Marks]
d) Write a method display_device_info() to show the device's name, model, and status. [3 Marks]
e) Handle cases where the device is already turned on when attempting to turn it on. [3 Marks]

Question 24 (20 Marks)

a) Construct the following classes: i. **Teacher** class with attributes name, subject, and years_of_experience. [2 Marks each]
ii. **Classroom** class with attributes class_name and capacity. [2 Marks each]
b) Write a method assign_teacher(teacher) in Classroom class to assign a teacher to a class. [5 Marks]
c) Create two classroom objects and assign teachers to them. [5 Marks]
d) Write a method display_classroom_info() to show the classroom's name, capacity, and assigned teacher. [3 Marks]
e) Handle cases where assigning the same teacher to multiple classrooms is not allowed. [3 Marks]

Question 25 (20 Marks)

a) Construct the following classes: i. **Artist** class with attributes name, genre, and number_of_albums. [2 Marks each]
ii. **Song** class with attributes title, duration, and artist (inheriting from Artist). [2 Marks each]
b) Write a method add_song(title, duration) in Song class to add a song. [5 Marks]
c) Create two song objects and assign them to artists. [5 Marks]
d) Write a method display_song_info() that shows the song's title, artist, and duration. [3 Marks]
e) Handle cases where a song is added without specifying its duration. [3 Marks]

Question 26 (20 Marks)

a) Construct the following classes: i. **Course** class with attributes course_name, course_code, and credits. [2 Marks each]
ii. **Instructor** class that includes an attribute courses_taught (a list of Course objects). [2 Marks each]
b) Write a method assign_course(course) in Instructor class to assign a course to an instructor. [5 Marks]
c) Create an instructor object and assign them multiple courses. [5 Marks]
d) Write a method display_instructor_info() to show the instructor's details and courses they teach. [3 Marks]
e) Handle cases where the same course is assigned more than once. [3 Marks]

Question 27 (20 Marks)

a) Construct the following classes: i. **Product** class with attributes product_name, price, and stock. [2 Marks each]
ii. **Order** class that includes attributes order_number and ordered_products (a list of Product objects). [2 Marks each]
b) Write methods add_product(product) and remove_product(product) in Order class. [5 Marks]
c) Create an order object and add products to it. [5 Marks]
d) Write a method display_order_info() to show the order number and products in the order. [3 Marks]
e) Handle cases where a product is removed from an order that doesn't contain it. [3 Marks]

Question 28 (20 Marks)

a) Construct the following classes: i. **Customer** class with attributes name, email, and loyalty_points. [2 Marks each]
ii. **Purchase** class with attributes purchase_id and amount (inheriting from Customer). [2 Marks each]

b) Write methods make_purchase(amount) and add_loyalty_points(points) in Purchase class. [5 Marks]
c) Create two customer objects and simulate purchases for both. [5 Marks]
d) Write a method display_customer_info() to show customer details and loyalty points. [3 Marks]
e) Handle cases where a purchase is made with insufficient loyalty points. [3 Marks]

Question 29 (20 Marks)

a) Construct the following classes: i. **Gadget** class with attributes gadget_name, brand, and price. [2 Marks each]
ii. **Laptop** class that inherits from Gadget and includes attributes screen_size and ram. [2 Marks each]
b) Write a method upgrade_ram(new_size) in Laptop class to upgrade the RAM. [5 Marks]
c) Create a laptop object and simulate upgrading its RAM. [5 Marks]
d) Write a method display_gadget_info() to show laptop details, including RAM size. [3 Marks]
e) Handle cases where RAM is upgraded to an invalid size (e.g., less than current size). [3 Marks]

Question 30 (20 Marks)

Construct the following classes: i. **House** class with attributes address, num_rooms, and price. [2 Marks each]
ii. **RealEstateAgent** class that includes an attribute listed_houses (a list of House objects). [2 Marks each]
b) Write methods list_house(house) and remove_house(house) in RealEstateAgent class. [5 Marks]
c) Create an agent object and list two houses. [5 Marks]
d) Write a method display_agent_info() to show agent details and the houses they have listed. [3 Marks]
e) Handle cases where a house is removed that wasn't listed. [3 Marks]

Question 31 (20 Marks)

a) Create an abstract class **Appliance** with abstract methods turn_on() and turn_off(). [2 Marks each]
b) Derive two concrete classes **WashingMachine** and **Refrigerator** from Appliance, implementing the abstract methods. Each class should have additional attributes (e.g., **WashingMachine**: load_capacity, **Refrigerator**: temperature). [3 Marks each]
c) Create instances of both classes and simulate turning them on and off. [5 Marks]
d) Write a method display_status() that shows whether each appliance is on or off and its unique attributes. [3 Marks]
e) Handle cases where an appliance is already on/off when trying to perform the action. [3 Marks]

Question 32 (20 Marks)

You are designing a game with different types of characters: a) Create an abstract class **GameCharacter** with attributes name and health, and abstract methods attack() and defend(). [5 Marks]
b) Derive classes **Warrior** and **Mage** that implement attack() and defend() differently. Each should have unique attributes like **Warrior**: strength, **Mage**: mana. [5 Marks]
c) Design an open-ended method for the GameCharacter class that can be used to interact with the environment (e.g., interact_with_object(object)), and implement this method in both derived classes. [5 Marks]
d) Create instances of both characters and simulate them attacking, defending, and interacting with an object. [5 Marks]

Question 33 (20 Marks)

a) Design an interface **PaymentMethod** with methods process_payment() and refund_payment(). [5 Marks]
b) Create two classes **CreditCardPayment** and **PayPalPayment** that implement the PaymentMethod interface, each with their own version of the methods. [5 Marks]
c) Allow students to add additional attributes and methods to these classes (e.g., credit card number, PayPal email) based on the scenario of their choice. [5 Marks]
d) Create instances of both payment methods, process a payment and refund for each, and display transaction details. [5 Marks]

Question 34 (20 Marks)

A company needs a system for managing employees of various types. a) Create an abstract class **Employee** with attributes name, employee_id, and salary, and an abstract method calculate_bonus(). [5 Marks]
b) Derive classes **Manager** and **Engineer** that implement calculate_bonus() differently. For example, managers may have a higher bonus percentage. [5 Marks]
c) Allow students to design their own additional methods for tracking project assignments and work hours, encouraging them to think creatively. [5 Marks]
d) Create instances of both employee types, calculate bonuses, and display employee details including projects or hours worked. [5 Marks]

Question 35 (20 Marks)

You are building a system for managing vehicles in a rental service. a) Create an abstract class **Vehicle** with attributes license_plate and mileage, and an abstract method calculate_rental_fee(). [5 Marks]
b) Derive classes **Car** and **Bike**, implementing calculate_rental_fee() based on mileage, type, or duration. [5 Marks]
c) Let students design their own methods for tracking rental history and availability status of vehicles. [5 Marks]

d) Create instances of both vehicle types, calculate rental fees, and display rental history and availability. [5 Marks]

Question 36 (20 Marks)

A media company manages different types of digital content. a) Design an abstract class **Media** with attributes title and release_date, and an abstract method play(). [5 Marks]
b) Derive classes **MusicTrack** and **Movie**, each implementing the play() method in its own way. MusicTrack should have attributes such as artist, while Movie should have attributes such as director. [5 Marks]
c) Let students define their own additional methods for handling media features like rating and bookmarking. [5 Marks]
d) Create instances of both media types, play the content, and use the additional methods to display extra features. [5 Marks]

Question 37 (20 Marks)

You're building a shopping application that supports different types of discounts. a) Create an abstract class **Discount** with an abstract method apply_discount(price) that returns the discounted price. [5 Marks]
b) Implement two classes **PercentageDiscount** and **FixedAmountDiscount** that extend Discount and apply discounts in different ways. [5 Marks]
c) Allow students to design their own methods for calculating and displaying the total price of items in a shopping cart, considering multiple discounts. [5 Marks]
d) Create instances of both discount types, apply them to some items, and display the final price after discounts. [5 Marks]

Question 38 (20 Marks)

You are tasked with creating a system to simulate animals in a zoo. a) Create an abstract class **Animal** with attributes species and age, and an abstract method make_sound(). [5 Marks]
b) Derive classes **Lion** and **Elephant**, each implementing make_sound() in a way that suits the animal (e.g., lion roars, elephant trumpets). [5 Marks]
c) Allow students to create their own open-ended methods for simulating interactions with zoo visitors or zookeepers, encouraging creative designs. [5 Marks]
d) Create instances of both animal types, simulate interactions, and display the animal's age, species, and sound it makes. [5 Marks]

Question 39 (20 Marks)

You're designing a system for managing different types of transport systems. a) Create an abstract class **Transport** with attributes name, capacity, and an abstract method calculate_speed(distance, time). [5 Marks]

b) Implement two classes **Bus** and **Train** that extend Transport and calculate speed based on different metrics (e.g., traffic for buses, number of stops for trains). [5 Marks]
c) Allow students to design additional methods that manage things like route planning or seat reservation systems. [5 Marks]
d) Create instances of both transport types, calculate their speeds, and manage additional features such as seat reservations. [5 Marks]

Question 40 (20 Marks)

A software company is creating a system for managing user roles in a web application.
a) Create an abstract class **User** with attributes username and password, and an abstract method `login()`. [5 Marks]
b) Implement two classes **AdminUser** and **GuestUser** that extend User, each implementing `login()` differently (e.g., AdminUser has additional verification steps). [5 Marks]
c) Encourage students to design their own methods for managing user actions like viewing pages, editing content, or handling permissions based on roles. [5 Marks]
d) Create instances of both user types, simulate logging in, and demonstrate different actions for AdminUser and GuestUser based on their roles. [5 Marks]