COMSATS University Islamabad
Department of Computer Science
Midterm Examination SPRING 2025

Class: BCS, BSE, BAI, BDS, BCT
Subject: CSC241- Object Oriented Programming
Instructor: Ms. Bushra Naz

Marks: 25
Time: 90 Mins
Date: 18-April-2025

Instructions:

- All questions are self-explanatory and require no further explanations during exam time.
- **Return the question paper along with the answer sheet.**
- **Attempt all questions.**

[CLO1: Demonstrate fundamental principles and concepts of object-oriented programming.]

Question 1. For the following Java snippet conclude and justify the output **[Marks-3]**

```
class Student {
    String name;
    int[] marks;

    Student(String name, int[] marks) {
        this.name = name;
        this.marks = marks;
    }

    int[] modifyArray() {
        int[] newMarks = new int[marks.length];
        for (int i = 0; i < marks.length; i++) {
            if (i % 2 == 0) {
                newMarks[i] = marks[i] * 2;
            } else {
                newMarks[i] = marks[i];
            }
        }
        return newMarks;
    }

    static void display(String name, int[] array) {
        System.out.println("Modified marks of " + name + ":");
        for (int mark : array) {
            System.out.print(mark + " ");
        }
        System.out.println();
    }
}
```

```
public class Main {
    public static void main(String[] args) {
        Student[] students = new Student[2];

        students[0] = new Student("Ali", new int[]{70, 80, 90});
        students[1] = new Student("Sara", new int[]{60, 75, 85, 95});

        for (Student s : students) {
            int[] modified = s.modifyArray();
            Student.display(s.name, modified);
        }
    }
}
```

Question 2. Provide short answers to the following questions. (Max 2-3 lines) [Marks-2*3]

a. Differentiate between Object class and an object.
b. Differentiate between public, private, protected, and default access modifiers.
c. Differentiate between abstraction and encapsulation.

[CLO 2: Apply the concepts of object-oriented programming principles along with interfaces and exception handling to solve a real-world problem.]

Question 3. A software company is developing a ride-sharing in which users can register as either passengers or drivers, both having attributes such as name, phone number, and email. Passengers can manage payment methods and maintain a ride history, whereas drivers are required to register one or more vehicles they operate, each with a registration number, model, capacity, and type (e.g., Car, Bike). Every user in the system can log in, update their profile, and rate other users. Both passengers and drivers have a rating, which is the average of feedback received from other users after completed rides. The feedback includes a score, a comment, and a timestamp. Drivers and passengers can rate each other mutually at the end of a ride.

A ride request is created by a passenger, specifying the pickup and drop-off locations, ride type preference, and the desired payment method. The system matches the request to an available driver based on location and vehicle type. Once the driver accepts, the ride becomes active, and details such as fare, distance, estimated time, and route are stored. A completed ride can then be reviewed and paid.

Based on the above description, draw a complete UML class diagram representing your object-oriented model. [Marks-6]

Question 4. Design a Java-based system for managing deliveries in an e-commerce platform. The system must support two types of products: physical and digital. Physical products include attributes such as product ID, name, price, weight, and dimensions, while digital products include product ID, name, price, download link, and license key. Customers can place orders containing multiple items, and each order is associated with a delivery partner (e.g., FedEx, UPS). While delivery partners can manage multiple orders, the system must preserve order data even if a delivery partner is removed. A shopping cart feature allows temporary holding of products during the customer's shopping process. Removing a product from

the cart does not remove it from the product catalog. The system must include the ability to calculate shipping weight (for physical products only) and the total price of a shopping cart. Additionally, when an order is canceled, all associated data (such as order items) must be deleted, but deleting a delivery partner must not affect orders that were previously assigned to them.

For the above scenario implement the following: **[Marks-10]**

a) **Implement a method to calculate the total shipping weight of an order:**
   - How can we compute the combined shipping weight by summing the weight of only physical products within an order?
   - Should digital products be excluded automatically from the shipping weight calculation?
   - What class should this method be implemented in, and how should it interact with OrderItem?

b) **Implement a method to calculate the total price of items in the shopping cart:**
   - How should we iterate through the shopping cart to calculate the total price, considering both physical and digital products?
   - Should discounts, taxes, or promotions be considered, or is it a raw sum of prices?
   - In what class should this method reside?

c) Lifecycle Rules:
   - What mechanism will be used to cancel an order and ensure all associated items are removed?
   - When deleting a delivery partner, how can we ensure that their assigned orders still persist correctly in the system?
   - Should we implement soft deletion or any other persistence strategy for delivery partners?

Good Luck ☺