National University

of Computer & Emerging Sciences Peshawar Campus

Student Name: _____ Roll No: _____

Program: BS(SE/CS)
Semester: Spring-2023
Time Allowed: 3 hours
Course: Software Design & Architecture (SE2002)
Software Design & Analysis (CS3004)
Object Oriented Analysis & Design (CS3031)

Examination: Final
Total Marks: 100 Weightage: 50%
Date: 26 May 2023
Instructor Name: Usama Musharaf

NOTE: Attempt all questions.

Question # 1: CLO-1 **Marks: [10] Time: [15 mins]**

a. Briefly discuss the steps involved in agile development.
b. Discuss the following (max 3 to 4 lines)
1. Software Views 2. Viewpoints 3. Liskov Substitution Principle 4. CRC cards

Question # 2: CLO-3 **Marks: [15] Time: [30 mins]**

Online Shopping System:

Online Shopping is a web-based application intended for searching, viewing, and selecting a product easier. It contains a sophisticated search engine for the users to search for products specific to their needs. The search engine will provide the available products based on the user's input. The user can then view the complete specification of each product. They can also view the product reviews and write their reviews as well. Customers can order products on the site and pay using different payment options like a credit card or a debit card. When a customer provides their card details, we want to validate it, run it through a third-party fraud detection system, and then send the details to a payment gateway for processing.

Design the online shopping system with the following constraints:

- Each class should have a single responsibility.
- Behaviour of the system should be open for extension but closed for modifications so that in the future new functionalities can be easily added.
- Follow proper design principles (when required) such as polymorphism, composition, etc.

Question # 3: CLO-3 **Marks: [15] Time: [30 mins]**

State Design Pattern allows an object to change its behavior when its internal state changes. By considering the scenario of an online shopping system (***given in Q # 1***), the object of ***order class*** can exist in different states which are as follows:

1. **Order Initial State:** In the initial state the order is just being placed, so no amount will be charged for the cancellation of an order in this state.
2. **Order Processing State:** This state indicates that the order is processed and ready to dispatch from the warehouse. An amount of 10% will be charged on order cancellation at this state.
3. **Order Dispatching State:** This state means that the order has been dispatched for delivery. An amount of 30% will be charged on order cancellation at this state.

Your task is to

- Draw a class diagram (with attributes and functions) using a state design pattern.
- Comment whether the state design pattern follows the open-close principle or not.

Question # 4: CLO-3	Marks: [15] Time: [30 mins]

Consider the following Problem Statement:

Suppose we have an employee record system in which we have different types of employees i.e. trainee employees (on probation) and some of the regular employees (salaried and hourly based). Consider the following design given below in ***Figure-1,*** it can be observed that

- Salaried Employee has a fixed amount of salary.
- Hourly Employee gets payment on the basis of the total hours he/she worked in a month.
- Trainee Employee gets payment in the form of a commission based on the items that he/she sold in a month e.g. 10$ for one product.
- Employee class has an abstract method named earnings (calculating earnings of the employee). All the derived classes have to provide their implementation.

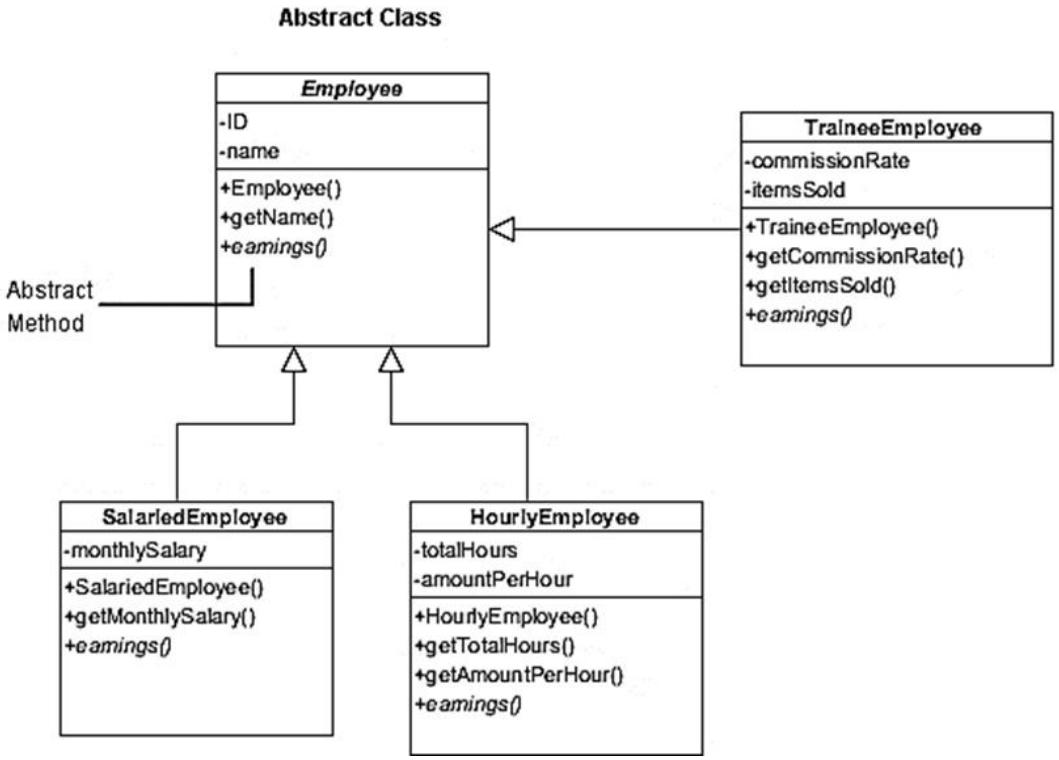

Figure 1 Class Diagram

Everything seems satisfactory until we think about the life of a trainee employee object. Trainee Employees can become regular employees after the probation period. If we delete the trainee employee object and create a new object of salary or hourly-based employee, then we have to destroy all the previous associations and links of that object as well which is a maintenance nightmare. This is known as object morphing when an object wants to change its behaviour.

- Re-Structure the above design to avoid maintenance issues and object morphing. Also, write some justification about your design strategy.
  Hint: Use Behavioral Design Patterns.

Question # 5: CLO-4	Marks: [15] Time: [30 mins]

ATM System:

A local bank intends to install a new automated teller machine (ATM) to allow users (i.e., bank customers) to perform basic financial transactions. ATM users should be able to view their account balance, withdraw cash (i.e., take money out of an account) and deposit funds (i.e., place money into an account). Consider the class diagram in ***Figure 2*** for your reference:

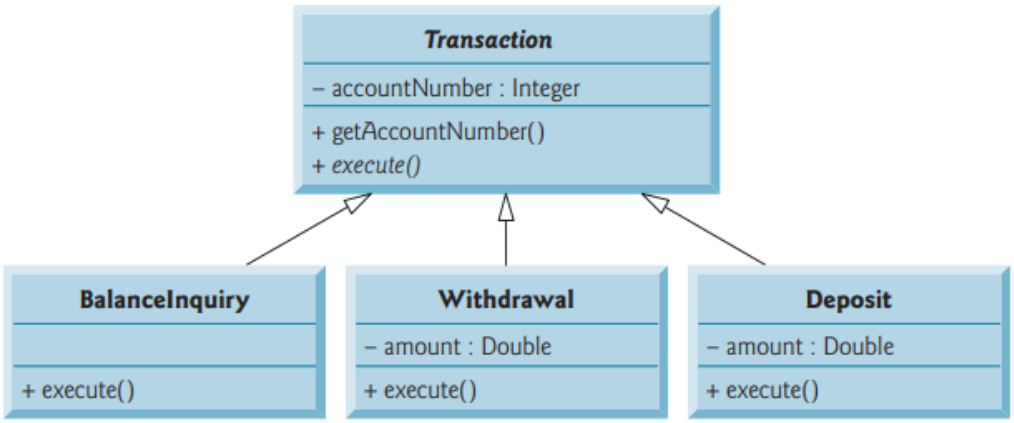

Figure 2 Class Diagram

- ***Your task*** is to write a java code by using Factory Design Pattern for the delegation of object creation to achieve loose coupling.
  Note: You may make your own intuition about methods/functions.

Question # 6: CLO-4	**Marks: [10] Time: [15 mins]**

Singleton Design Pattern ensures that there is only one instance of a class and provides a global point of access to it. Consider the scenario of an ATM transaction given in ***Question # 6*** and write a java code to restrict the creation of multiple instances of transaction class.

Question # 7: CLO-5	**Marks: [10] Time: [15 mins]**

Fire Alarming System: The owner of a large multi-stored building wants to have a computerized fire alarm system for his building. Smoke detectors and fire alarms would be placed in each room of the building. The fire alarm system would monitor the status of these smoke detectors. Whenever a fire condition is reported by any of the smoke detectors, the fire alarm system should determine the location at which the fire condition is reported by any of the smoke detectors, and then sound the alarms only in the neighboring locations. The fire alarm system should also flash an alarm message on the computer console.

- Design an architecture based on an event-driven style.

Question # 8: CLO-5	**Marks: [10] Time: [15 mins]**

- Discuss microservices architecture by comparing it with a monolithic architectural approach.

GOOD LUCK!