**Advanced Programming (CSE201), Quiz -1**

**Time allocated: 03:20pm – 3:45pm (25 minutes) + 5 minutes for uploading solution**

**Instructions:**

· You must follow all the instructions sent to you earlier over the email.

· Only reasonable and clearly mentioned assumptions (if any) would be accepted.

· For justifications, please be as concise as possible (2-3 sentences only)

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Q1)** Read the below mentioned problem description and answer below questions:

“*An electronic shop repairs the motherboard of Laptops and Tablets and replaces the tempered glass of Smartphones and Tablets. This shop has a mechanic responsible for receiving the Laptops, Tablets, and Smartphones; storing them over two different boxes inside the electronic shop; and for providing the services. These boxes are essentially storing Laptop, Tablet, and smartphones according to the requested service. One of the boxes can store Laptop and Tablet to repair their motherboard, whereas the other can store Smartphones and Tablets for replacing the tempered glass. For repairing the motherboard, the Laptop and Tablet must be shutdown. Shutdown in Laptop makes it hibernate, whereas the shutdown in Tablet simply turns it off. For replacing the tempered glass, the airplane mode must be activated on Tablets and Smartphones. Airplane mode in Tablet switches off wifi whereas in Smartphones it will switch off the mobile network. The mechanic receives the Laptop, Tablet, and Smartphones throughout the day and then services them together by taking them out of their respective boxes. Any Laptop, Tablet, and Smartphones can be given for only one type of service.*”

Implement the Object Oriented Implementation of the above program description and identify the class relationships (if any). No need to code the main method. You should only use the concepts taught in Lectures up to interfaces and polymorphism. You must write actual code (no pseudocode). **[7 marks]**

Rubric for Q1: (Total 7 marks)

No need to see the working code. Only see the below points:

1. Classes Shop and Employee
   1. +0.5 marks
2. Data encapsulation being followed inside each classes Shop and Employee
   1. +0.5 marks
3. Two List (or ArrayList) type objects inside Shop class as List<InterfaceA> and List<InterfaceB> (Polymorphism in parameter)
   1. +0.5 marks
4. If Shop instantiates Employee then composition relationship, else association relationship / If employee inherits the shop class
   1. +0.75 marks
5. The employee has association relationship with Shop / If the employee has protected access of shop class variables or has their access through a getter function
   1. +0.5 marks
6. InterfaceA/Abstract\_classA with method declaration “public void serviceA()”
   1. +0.5 marks
7. InterfaceB/Abstract\_classB with method declaration “public void serviceB()”
   1. +0.5 marks
8. Three things come for service (classes TypeX, TypeY, and TypeZ)
   1. +0.75 marks
9. class TypeX implements/extends InterfaceA/Abstract\_classA and provide concrete implementation of serviceA
   1. +0.5 marks
10. Class TypeY implements/extends InterfaceA/Abstract\_classA and InterfaceB/Abstract\_classB and provide concrete implementations of methods serviceA and serviceB
    1. +1 marks
11. Class TypeZ implements/extends InterfaceB/Abstract\_classB and provide concrete implementation of serviceB
    1. +0.5 marks
12. All overridden methods annotated with @Override
    1. +0.5 marks

**Q2)** Correct the following program. It must follow all principles/coding practices of OOP **[3 marks]**

public class Main{

public static void main(String[] args){

/\* If some student removes the parameterized constructor in Elephant then NO need to pass String type name here Hence, marking scheme as follows for below

1. If parameterized constructor removed from Elephant

OR

1. String name passed here correctly
   1. +1 marks

\*/

Animal A1 = new ~~Elephant();~~ Elephant(“abc”)

**OPTION-A**

~~Animal A2 = new Animal ();~~

/\* Elephant on both LHS and RHS: NO MARKS HERE

Elephant A2 = new Elephant(“def”);

**OPTION-B**

Animal A2 = new Elephant(“def”); //only string passed as parameter

A1.eat();

A1.dance(A2);

}

}

public interface Animal {

public void eat();

**OPTION-A**

~~private~~ public void dance(~~Animal~~ Elephant A); // +1 marks

**OPTION-B**

~~private~~ public void dance(Animal A); // +1 marks

}

public class Elephant implements Animal {

private String name;

public Elephant (String n){name = n; }

public void eat(){

System.out.println(“Elephant Eating”);

}

**OPTION-A**

public void dance(~~Animal~~ Elephant A){ // +1 marks

System.out.println(this.name+” dancing with “ +A.name);

}

**OPTION-B**

public void dance(Animal A){

System.out.println(this.name+” dancing with “ +(( Elephant)A).name); // +1 mark

}

}