Question 1 – Ethics (5 Marks):

Cheating in the Closed-book Online Examination.

During the current coronavirus pandemic, CityU has decided to run all final examination online, and trust that their students are capable of supervising themselves and respecting the system while carrying out online examination without having any academic misconduct, such as cheating by seeking for solutions from any other available materials including the Internet, Books, and any related Teaching materials.

A student X studying in CS3342 Software Engineering Design, fully aware of the examination rules and arrangements and its closed-book nature. X is gaining unfair advantage by searching Google and other Internet online resources, and obtained solutions required for the exam questions.

Another student Y also studying in CS3342 Software Engineering Design, pre-arranged a study room on campus and teaming up with student X above to working on the examination together, discussion and working towards pre-divided examination questions with student X.

By taking unfair advantage of other honest students in such a difficult coronavirus pandemic, in your opinion

(a) Are student X and Y ethical, and why? (1 Mark)

(b) If you happens to have observed such cheating misconduct, and without reporting to the course lecturer or a concerning University staff, are you ethical and why? (1 Mark)

Please assess the above ethical concerns by filling in the following table : (3 Marks)

Code of Software Engineering Ethics	**Are the Student X and Y Ethical?**	**If you know and do not report, are you ethical?**
Public interest		
Client and Employer		
Product		
Judgement		
Management		
Profession		
Colleagues		
Self		

Question 2 – Software Engineering in General (10 Marks):

(a). In your own words, please explain what is “**Software Engineering**”. (2 Marks)

(b). In your own words, explain what is “**Software Development Process**” (also known as Software Development Life Cycle or Software Process). (2 Marks)

(c). In your own words, explain what is “**Functional Requirement**”, you also need to provide an example in reference to your own group project. (3 Marks)

(d). In your own words, explain what is “**Component Based Software Engineering**” and what are the benefits of CBSE? (3 Marks)

Question 3 – Roles of Variables, Sequence Diagrams and Requirements (30 Marks):

(a) Study the following Java codes and identify the role of each variable in the table below.
You may use Table 1 for your reference.

Figure 1. *Seller.java and Goods.java*

```java
import java.util.*;

public class Seller {
    private List<Goods> goods_list = new ArrayList<Goods>();
    private final double discount_rate = 0.8;

    public void addGoods(int num) {
        Scanner scanner = new Scanner(System.in);
        for (int i = 1; i <= num; i++) {
            String goods_name = scanner.next();
            Goods goods = new Goods(goods_name);
            goods_list.add(goods);
        }
    }

    public double calculateRevenue() {
        double sum_revenue = 0.0;
        for (Goods goods : goods_list) {
            sum_revenue = sum_revenue + goods.getRevenue();
        }
        double final_revenue = sum_revenue * discount_rate;
        return final_revenue;
    }

    public double getMaxRevenue() {
        double max_revenue = 0.0;
        double current = 0.0;
        boolean neg_revenue = false;
        final int m = goods_list.size();
        for (int i = 0; i < m; i ++) {
            current = goods_list.get(i).getRevenue();
            if (current < 0) {
                neg_revenue = true;
                System.out.println("Error exists.");
            }
            if (current > max_revenue) {
                max_revenue = current;
            }
        }
        return max_revenue;
    }
}

public class Goods {
    private String goods_name;
    private int  sales;
    private int price;
    public Goods(String name) {this.goods_name = name;}
    public double getRevenue() {return price * sales;}
}
```

Table 1: Roles for Variables in Software Programs

Role	**Description**
Constant/ Fixed value	A variable which is initialized without any calculation and whose value does not change thereafter.
Stepper	A variable stepping through values that can be predicted as soon as the succession starts.
Most-recent holder	A variable holding the latest value encountered in going through a succession of values.
Gatherer	A variable accumulating the effect of individual values in going through a succession of values.
Transformation	A variable that always gets its new value from the same calculation from value(s) of other variable(s).
One-way flag	A two-valued variable that cannot get its initial value once its value has been changed.
Temporary	A variable holding some value for a very short time only.
Organizer	A data structure, which is only used for rearranging its data and object elements after initialization.

Your Answer:

Variable	**Role of Variable (1 Mark Each)**
goods_list	
discount_rate	
num	
i	
goods_name	
sum_revenue	
final_revenue	
current	
neg_revenue	
m	

(c) Based on Figure 1. *Seller.java,* draw a complete **Sequence Diagram** using Visual Paradigm for the function **public void** addGoods(**int** num). (Hint: You may use the reverse generation feature in VP)
(5 Marks)

(d) Based on Figure 1. *Seller.java,* draw a complete **Sequence Diagram** using Visual Paradigm for the function **public void** addGoods(**int** num). (Hint: You may use the reverse generation feature in VP)
(5 Marks)

(e) **Requirement Specifications (10 Marks)**

In order to further define functional requirements for the Customer and Checkout scenario for the system, the following defines the requirement specifications for the **Checkout** use case.

After the **Customer** selected the goods and ready for **Checkout** (use case), which (1) it will **Calculate Total Amount**, and then (2) proceed to the **Payment**, where **Customer** will be given two options for **Payment**:

a. **Pay by Cash**, the system will accept Cash.
   or
b. **Pay by Gift Card**, the system will deduct the amount from the shopping gift card.

The system will verify and check only IF the payment is successfully completed, then a **Receipt is Issued** to the customer, and the customer can take the receipt and the goods purchased. IF unsuccessful, an error will be displayed to Customer.

Complete the following use case specification table for the **Checkout** use case under typical course of events (assuming customer pay by Cash) AND alternative course of events (assuming customer pay by Gift Card). This situation involves the **Customer** actor.

Use Case Name:	**Checkout**	
Actor(s):	Customer	
Description:	This use case describes the process of a customer completing the checkout for the goods selected. On successful completion, a receipt will be issued.	
Reference ID:	SHOP-1.1	
Typical course of events:	Actor Action	System Response
	Step 1: This use case is initialized when a customer proceeds to pay for goods selected.	**Step 2:**
	Step 3: The Customer chooses to pay by Cash.	**Step 4**:
		Step 5:
	Step 6:	
Alternative course of events:		
Pre-condition:		
Post-condition:	The completed transactions will be recorded.	

Question 4 – Object Oriented Software Design - CILO3 (25 Marks)

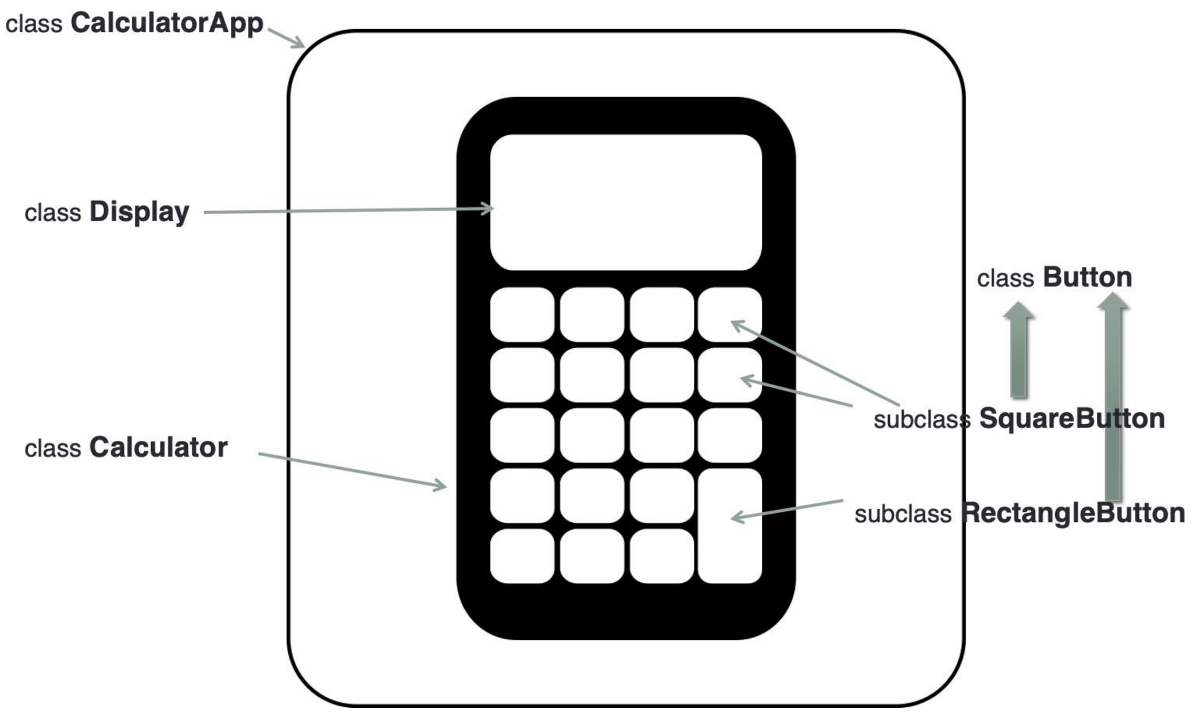

Figure 2. Calculator Design

Object Inheritance

(a) In your own words, explain what is “**Inheritance**” in Object-Oriented Programming Design. (3 Marks)

(b) Provide an example from the above Calculator design, that **Inheritance** could be used. (3 Marks)

Metamorphism

(c) In your own words, explain what is “**Metamorphism**” in Object-Oriented Programming Design. (3 Marks)

(b) Provide an example from the above Calculator design (Figure 2), that **Metamorphism** could be used. (3 Marks)

Class Association Linkage and Multiplicity

(e) There exist three main types of Class Linkages, namely **Composition, Aggregation and Association**. Which is the most appropriate relationship between class Calculator and Button (Figure 2), justify your choice. (2 Marks)

(f) What is the most appropriate Multiplicity to describe the Class Association between Calculator and Button(Figure 2), justify your answer. (2 Marks)

(g) There is only one Display attached to the Calculator(Figure 2), which Design Pattern could be used? (2 Marks)

(h) Using Visual Paradigm to draw a Class Diagram to illustrate the Calculator Design (Figure 2), including correct association and multicity (attributes and operators ARE NOT REQUIRED, but optional) (7 Marks)

Question 5 – Software Design Principles and Design Patterns - CILO3 (30 Marks)

Project: Bus Ticketing
Based on the **BusTicket.java** code listings (see below), study the function ***calculateFare (),*** *this is designed to correctly calculate the Bus ticket fares for three different Bus Zones, namely* **CITY, INTER_CITY and INTER_STATE.**

Bus Fare Table

AREA (zone)	Single Ticket	Return Ticket
CITY	$8.0	$10.0
INTER CITY	$16.0	$20.0
INTER COUNTRY	$30.0	$50.0

Figure 3. BusTicket.java

```
public class BusTicket
{
    private String BusZone;
    private boolean isReturnTicket;
    private String Title;
    private String Name;
    private int birthDay;
    private int birthMonth;
    private int birthYear;

    BusTicket(boolean returnTicket, String _title, String _name,
            int _birthDay, int _birthMonth, int _birthYear, String _busZone)  {
        isReturnTicket = returnTicket;
        Title = _title;
        Name = _name;
        birthDay = _birthDay;
        birthMonth = _birthMonth;
        birthYear = _birthYear;
        BusZone = _busZone;
    }

    public int calculateFare () {
        double fare = 0;

        if (BusZone == "CITY") {
            if (isReturnTicket)
                fare = 10;
            else
                fare = 8;
        } else if (BusZone == "INTER_CITY") {
            if (isReturnTicket)
                fare = 20;
            else
                fare = 16;
        } else if (BusZone == "INTER_STATE") {
            if (isReturnTicket)
                fare = 50;
            else
                fare = 30;
        }
            return fare;
    }

    public void changeBusZone(String _busZone) {
        BusZone = _busZone;
    }
}
```

(a) Based on *Figure 3.* BusTicket.java, using Visual Paradigm to draw a Full class diagram. **(5 Marks)**
Note: your solution must include all the correct attributes, data types, constructor, and operations/functions listed in the code accordingly.

(Only UML class diagram produced by **Visual Paradigm** is acceptable, place your diagram below)

Based on the *Figure 3.* BusTicket.java, answer the following and create a New Design:

(b) The class constructor ***BusTicket(…)*** contains a long list of parameters describing the attributes of a **Customer** buying the ticket, and it would be difficult to maintain the code and design if would place them in a single class. This is in violation of ________________________ Design Principle. **(2 Marks)**

(c) To address the above violation in (a) and to provide a better design, you must Redesign the class diagram and to include a new class called **Customer** if necessary. **(8 Marks)**

- ➔ Place your design in next Page.

(d) The class function **calculateFare()** is designed to correctly calculate the Bus ticket fares for three different Bus Zones, namely **CITY, INTER_CITY and INTER_STATE,** and the fare is also calculated according to the Boolean value **isReturnTicket.** According to *Figure 3.* BusTicket.java, unfortunately this design do not allow its behaviour to be further extended without modifying the source code of BusTicket.java. In Software Design Principle, this could be described as potential violation of ____________________ Design Principle, making addition of new Bus Zones a difficult task in the future.

(2 Marks)

(e) To address the above violation in (d), what Design Pattern would be appropriate? **(2 Marks)**

(f) In addition to (e), using the most appropriate Design Pattern to Redesign the UML class diagram.

(11 Marks)

- ➔ Place your design on the next Page.

Redesign Project Bus Ticket. According to (c) and (f), redesign project Bus Ticket, including necessary changes in function/operation rearrangements or relocations according to the best practices in object-oriented design in your new design. Your new design may need to include the BusTicket class and other related classes. Please observe the correct usage of Links (Association, Aggregation, Composition).

(Only UML class diagram produced by **Visual Paradigm** is acceptable, place your diagram below)

Section 5(c) 8 Marks + 5(f)11 Marks.

-- END --