Duration: 2 hours and 30 minutes
Aids Allowed: None

Student Number: ____________
Last Name: ________________________
First Name: _________________________
Instructor: _________________________

Do not turn this page until you have received the signal to start.
(In the meantime, please fill out the identification section above,
and read the instructions below carefully.)

This final consists of 4 questions on 17 pages (including this one),
printed on one side of the paper. Use a pen to mark the answers
to questions. Do NOT use pencil. When you receive the signal
to start, please make sure that your copy of the test is complete. If
you need more space, use the reverse side of the page and indicate
clearly the part of your work that should be marked.

MARKING GUIDE

# 1: _____/20
# 2: _____/20
# 3: _____/20
# 4: _____/20
TOTAL: _____/80

Good Luck!
Question 1.  [20 MARKS]
The following constitutes a set of simple questions that relate to the material covered in the lecture. Answer each one of them with a simple answer. Note, often a single word suffices as answer, or a simple sentence will do. Carefully read the question.

Part (a)  [6 MARKS]
List, at least, six distinct middleware platform categories and for each category, give one concrete example of a specific middleware platform. (E.g., if we were talking about programming languages, one correct answer would be object oriented programming languages and Java, as one representative example.)

Note, if the middleware category and middleware platform example you list do not correspond the full points will not be given for either one. (You do not need to explain !)

<table>
<thead>
<tr>
<th>Category</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td></td>
</tr>
<tr>
<td>2.</td>
<td></td>
</tr>
<tr>
<td>3.</td>
<td></td>
</tr>
<tr>
<td>4.</td>
<td></td>
</tr>
<tr>
<td>5.</td>
<td></td>
</tr>
</tbody>
</table>

Part (b)  [1 MARK]
Your task is to build a software solution that facilitates the exchange of information between \( n \) to \( m \) entities, where \( n \) and \( m \) are very large and don’t know about each other. What is a good middleware paradigm to use in this case?

Part (c)  [1 MARK]
Your task is to build a software solution to integrate a relational database, an object relational database, and object oriented database, and a set of web-sites. One requirement indicates that one unique interface should be used to access all information. What is a good middleware abstraction to do that?

Part (d)  [1 MARK]
What is the difference between an application server and a web server?
Part (e) [4 MARKS]
List three communication paradigms and discuss their characteristics with respect to the number of parties connected (a.k.a. arity), their decoupling in time (i.e., whether the communicating parties must be up at the same time), and their decoupling in space (i.e., whether the communicating parties know each other).

<table>
<thead>
<tr>
<th>Paradigm</th>
<th>Arity</th>
<th>Decoupled in Time</th>
<th>Decoupled in Space</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The following sub-questions refer to the following description:

The following pseudo code computes a scalar product (multiplication between a vector ‘b’ and a matrix ‘matrix’). The matrix is stored on a remote data server, called ‘server’.

```java
for (i=0; i++; i<N){
    for (j=0; j++; j<N){
        c[i] = b[j] * server.matrix(j,i);
    }
}
```

Part (f) [1 MARK]
In the above code, identify the piece of code that leads to a communication taking place.

Part (g) [2 MARKS]
From the code above, extract a possible interface of the remote server. Make sure you use all information given in the pseudo code snippet. Either, use Java interfaces, or a pseudo code notation to define the interface.

Part (h) [1 MARK]
Is the result stored on the remote server or the local client?
Part (i) [1 MARK]

How many communication exchanges (i.e., opening a connection, transferring data, and closing a connection) are taking place? Express your answer analytically. Hint: Assume that for each remote method invocation a separate connection has to be established.

Part (j) [1 MARK]

What is the least amount of communication exchanges (i.e., opening a connection, transferring data, and closing a connection) that have to take place for this computation to take place?

Part (k) [1 MARK]

Now assume that a large amount of overhead is associated with opening a connection and closing it. This overhead exceeds by far the time to do the actual data transfer and the simple multiplication done by the above code. How could this insight be used to optimize the above code snippet? Simply discuss a viable optimization.
Question 2. [20 marks]
In this question you have to model with UML. Make sure you stay as close to the original UML language as possible. If you don’t remember certain symbols, explain your symbolism.

Part (a) [10 marks]
Provide the UML class diagram for the following code. Note, the code listing shows several different files.

Please draw the individual class diagrams next to the code below. Draw the overall class diagram, showing all dependencies on the following page.

```
Draw individual class diagrams here:

// file Customer.java

public class Customer {
    private String name;
    private String address;
    private Order[] order;

} // end Customer

// file Order.java
import java.util.Date;

public class Order {
    private Date date;
    private String status;
    private Customer customer;
    private OrderDetail[] orderDetail;
    private Payment[] payment;

    public float calcTax() {
        ....
    } // end calcTax

    public float calcTotal() {
        ....
    } // end calcTotal

    public float calcTotalWeight() {
        ....
    } // end calcTotalWeight

} // end Order
```
// file Payment.java
public abstract class Payment {
    protected float amount;
    ...
} // end Payment

// file Cash.java
public class Cash extends Payment {
    private float cashTendered;
    ...
} // end Cash

// file Check.java
public class Check extends Payment {
    private String name;
    private long bankID;
    public boolean authorized() {
        ...
        ...
    } // end authorized
    ...
} // end Check

// file Credit.java
import java.util.Date;

public class Credit extends Payment {
    private long number;
    private String type;
    private Date expDate;
    public boolean authorized() {
        ...
        ...
    } // end authorized
    ...
} // end Credit

// file OrderDetail.java
public class OrderDetail {

private int quantity;
private boolean taxStatus;
private Order order;
private Item item;

public float calcSubTotal() {
    ...
} // end calcSubTotal

public float calcWeight() {
    ...
} // end calcWeight

} // end OrderDetail

// file Item.java
public class Item {
    private float shippingWeight;
    private String description;

    public float getPriceForQuantity() {
        ...
    } // end getPriceForQuantity

    public float getWeight() {
        ...
    } // end getWeight

} // end Item
Draw the overall class diagram, showing all dependencies found in the code, on this page. You may refer to each individual class diagram by its name (i.e., name of class in the code.)

**Part (b) [10 marks]**

For the following code provide the sequence diagram that describes the interactions between classes for a call to method `printFile` in class `Computer`:

```java
// file Computer.java
public class Computer{
    ...
    PrintServer printServer;
    ...

    public void printFile(File f){
        printServer.print(f);
    }
}
```
// file PrintServer.java
public class PrintServer {
    ...
    Printer printer;
    Queue printerQueue;
    ...

    public void print(File f) {
        if (printer.free())
            printer.print(f);
        else
            printerQueue.store(f);
    }
    ....
}

// file Printer.java
public class Printer {
    ...
    public void print(File f) {
    ...
    }
}

// file Queue.java
public class Queue {
    ...
    public void store(File f) {
    ...
    }
    ....
}
Question 3. [20 marks]

Assume that you are asked to implement a purchasing system for a gearbox store. The purchasing system allows gearbox customers to freely select components of the gearbox from any vendors. Suppose that the gearbox only consists of gears, bearings and shafts, which are the only parts carried by the store. Every part has a part number, which UNIQUELY identifies that part, a vendor name, a unit price and a unit weight. Individual parts can be assembled together to form complex parts. In our system, the gearbox consists of any number of gears, bearings and shafts. The unit price of the complex part is the sum of all its components. The purchasing system should compute the total price of any gearboxes that are assembled. The engineering team A has developed the UML diagram for the purchasing system.

The following is the documentation for the classes:

Part: An abstract class represents a gearbox part, which has a part no, a vendor name, a unit price and a unit weight. In addition to setters and getters of its properties, it also contains methods for assembling and disassembling sub-components. Note that the class does not provide any implementations of the assembling and disassembling methods.

Gear: A gear is defined by its diameter and the number of teeth it has.

Bearing: A bearing is defined by its color and its center thickness.
Shaft: A shaft is defined by its length and its diameter.
GearBox: A gearbox is assembled from any number of gears, shafts and bearings.

Part (a) [10 marks]

Your task is to provide the signatures of the classes in Java to EXACTLY implement the UML model presented. That means you do need to correctly provide the code for all attributes and methods (i.e., the signatures), but you Do NOT need to provide any implementation.

Part (b) [10 marks]

Provide a complete implementation of the Assemble and Disassemble methods of the GearBox class. If you can't remember the exact Java syntax, try to make your code as logically consistent as possible. You can ignore the package construct.

The following are a few methods in the java.util.Vector that you might find useful.

- //Adding an object to the vector
  void add(Object o);

- //Removing an object at the corresponding position in the vector as indicated by index
  void remove(int index);

- //Return number of elements in the vector
  int size();

- //Clear the elements of vector
  void clear();

You may also find the method instanceof useful. It is used to determine whether two objects are the same type by returning a boolean value. For example, you can write the following:

if (objectA instanceof ObjectB) {...}
Question 4.  [20 MARKS]
You are to specify a control system for elevators using UML. Here is the specification of the system:

The elevator system has several co-located elevators in a tall office building. The control system has to schedule elevators to respond to requests from users at various floors and control the motion of the elevators between floors. The control system has a centralized elevator controller to control the logic and behavior of the elevators. Each elevator has a set of doors, which is controlled by commands to open or close. When a door is opened, the elevator controller will start a timer. After 15 seconds have elapsed, the controller will send a message to the elevator door component to close the doors. Each elevator has a motor, which is controlled by commands to move or stop the elevator. On each floor, there are two floor buttons, an UP button and a DOWN button. A user presses a button to request an elevator. There is no DOWN button on the ground level, and there is no UP button on the top floor. Inside each elevator, there is a set of elevator buttons. Each elevator button has a corresponding lamp. A user presses an elevator button to select a destination floor. When an elevator button is pressed, the elevator controller is notified. The elevator controller is responsible for controlling when to turn on or off the lamps of the elevator buttons. The lamp is turned on if a button press is detected. The lamp is turned off after the elevator has reached the floor to be visited.

Informal specifications may be ambiguous. If you perceive any underspecification, over-specification or ambiguity in the above specification, make an assumption and clearly state your assumptions in your answer.

Part (a)  [10 MARKS]
Draw the class diagram to represent the elevator system as describe above. Your diagram should adequately indicate the components in the system, their properties and behavior. In your diagram, use inheritance where possible. Also show the cardinality between classes.

Hint: The relevant parts of the elevator system for you to model are the elevator buttons, the elevator controller, the elevator doors, and the elevator motor.

Write your answer here:
Part (b) [10 MARKS]
A use case of the elevator is given below in plain English. In this scenario, assume we only have one elevator. Draw the corresponding UML sequence diagram for this scenario.

Hint: First read all the steps below. Identify all system components that respond to messages.

A sequence diagram models a system’s behavior over time. Make sure you indicate where each of the steps described below intervene, i.e., label the message sent/receives with the actions listed below.

Scenario:

1. User A presses the Up floor button of floor 5 to request an elevator. User A wishes to go to floor 15.

2. The floor button informs the elevator controller that the floor button has been pushed.

3. The elevator controller sends a message to the Up floor button to turn itself on.

4. The elevator is currently on floor 2. The elevator controller sends a message to the motor of the elevator to move itself up to floor 5.

5. The elevator controller sends a message to the UP floor button to turn itself off.

6. The elevator controller sends a message to the elevator doors to open themselves.

7. The elevator controller starts a timer. (User A enters the elevator)

8. User A presses elevator button for floor 15.

9. The elevator button informs the elevator controller that the elevator button has been pushed.

10. The elevator controller sends a message to the elevator button for floor 15 to turn itself on.

11. The elevator controller sends a message to the elevator doors to close themselves after a timeout of the timer.

12. The elevator controller sends a message to the elevator motor to move the elevator to floor 15.

13. The elevator controller sends a message to the elevator button for floor 15 to turn itself off.

14. The elevator controller sends a message to the elevator doors to open themselves to allow User A to exit from the elevator.

15. The elevator controller starts a timer. (User A exits the elevator)

16. The elevator controller sends a message to the elevator doors to close themselves after a time out.
Write your answer on this page.