University of Ruhuna- Faculty of Technology

Bachelor of Information & Communication Technology Honours Degree
Level 3 (Semester I) Examination, June 2024
Academic year 2022/2023

Course Unit: ICT3112 Advanced Programming **Duration:2 hours**

- Answer all ***Four (04)*** questions.

1)

a)

i) List down three (03) reasons for using design patterns. **[15 marks]**

ii) Provide a scenario where you can apply each of the following design patterns in software development. **[25 marks]**

a. Strategy

b. Facade

c. Factory

d. Command

e. Observer

b) Consider a scenario where a drawing application can only work with shapes defined by a `Point` class (x, y coordinates). However, you want to integrate a legacy library that uses a `Rectangle` class (width, height, and optional origin point).

i) Explain how you would use the Adapter pattern to achieve this integration. **[20 marks]**

ii) What are the elements (i.e. classes, interfaces, relationships) of the UML class diagram that models the integration mentioned in Question 1 Part b (i). **[15 marks]**

c) State three (03) design patterns used in the following Java code (Figure 1). Furthermore, describe which classes play which roles in those patterns **[25 marks]**

```
public interface Navigator {
    public Route navigate();
}
public interface Route {
    public void addStop(Stop s);
    public void travel();
}
public interface Stop {
    public void visit();
}
public class NavigatorImpl implements Navigator {
```

```java
    private static NavigatorImpl instance = new NavigatorImpl();
    private NavigatorImpl() {}
    public static NavigatorImpl getInstance() {
        return instance;
    }

    public Route navigate() {
        return new RouteImpl();
    }
}

public class RouteImpl implements Route {
    private ArrayList<Stop> stops = new ArrayList<Stop>();
    public void addStop(Stop s) {
        stops.add(s);
    }
    public void travel() {
        for (Stop s : stops) {
            s.visit();
        }
    }
}

public class Place implements Stop {
    public void visit() {
        System.out.println("Visiting the place!");
    }
}

public interface Destination {
    public void reach();
}

public class DestinationAdapter implements Stop {
    private Destination destination;
    public DestinationAdapter(Destination destination) {
        this.destination = destination;
    }
    public void visit() {
        destination.reach();
    }
}

public class TouristSpot implements Destination {
    public void reach() {
        System.out.println("Reached the tourist spot!");
    }
}
```

Figure 1

2)

a)

i) Explain the following states of a thread **[20 marks]**

a. New

b. Runnable

c. Blocked

d. Waiting

e. Terminated

ii) The two methods shown in Figure 2 update the balance of a user and their bank account. What is the possible issue that occurs if both methods are run parallelly? Explain how you fix that issue. **[20 marks]**

Method 1

```
void updateAccount(BankAccount account){
    synchronized(account){
        User user = account.getUser();
        double tempBalance;
        synchronized(user){
            tempBalance = user.getBalance();
        }
        user.setBalance(account.getBalance());
    }
}
```

Method 2

```
void updateAccount(User user){
    synchronized(user){
        BankAccount account = user.getAccount();
        double tempBalance;
        synchronized(account){
            tempBalance = user.getBalance();
        }
        account.setBalance(user.getBalance());
    }
}
```

Figure 2

b)

i) Threads often need to pause and allow other threads to execute. Describe five (05) different Java methods by which a thread can yield control of the CPU to other threads. **[20 marks]**

ii) Inter-thread communication allows threads to coordinate their actions by notifying each other about changes in state. Briefly explain four (04) ways to implement inter-thread communication in Java. **[20 marks]**

c) Consider the Readers-Writers problem, where a set of reader and writer threads access a shared resource (e.g., a database). Readers can access the resource concurrently, but writers need exclusive access. Let the readers and writers be represented by separate threads. Each reader executes Read(i), and each writer executes Write(i), where "i" is the thread's number. Assume that there is a read-write lock that controls access to the shared resource. The pseudocode is given as: **[20 marks]**

```
class ReadWriteLock {
    private int readers = 0;
    private boolean writer = false;

    synchronized void startRead() {
        while (writer) {
                wait();
                Thread.currentThread().interrupt
        }
        readers++;
    }
    synchronized void endRead() {
        readers--;
        if (readers == 0) {
            notifyAll();
        }
    }
    synchronized void startWrite() {
        while (writer || readers > 0) {
                wait();
                Thread.currentThread().interrupt();
        }
        writer = true;
    }
    synchronized void endWrite() {
        writer = false;
        notifyAll();
    }
}
ReadWriteLock lock = new ReadWriteLock();

void read(int i) {
    lock.startRead();
    lock.endRead();
}

void write(int i) {
    lock.startWrite();
        lock.endWrite();

}
```

Figure 3

Assume that a deadlock occurs when executing the code shown in Figure 3. List the four (04) conditions of deadlock and explain why each of them is satisfied in this scenario.

3)

a)

i) What is the problem related to serialization in the code shown in the Figure 4? State how you fix it. **[20 marks]**

```
class ProblematicPizza {
    private ProblematicCheese cheese;

    public static void main(String[] args) {
        try {
            ProblematicPizza pizza = new ProblematicPizza();
            ObjectOutputStream objectOutputStream = new
ObjectOutputStream(new FileOutputStream(new File("pizzaFile.txt")));
            objectOutputStream.writeObject(pizza);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Figure 4

ii) In Java serialization, you have a class 'SensitiveData' with some fields that you do not want to serialize. Describe how you can prevent specific fields from being serialized when an object of this class is written to an 'ObjectOutputStream'. **[20 marks]**

b) Explain the differences between the following concepts.

i) Stub vs. Skeleton **[10 marks]**

ii) Connection-Oriented vs. Connection-Less protocols **[10 marks]**

c) For each of the following statements, indicate whether it is true or false. Justify your answers. **[40 marks]**

i) The java.lang.ThreadGroup class is a subclass of the java.lang.Thread class.

ii) The Thread.start() method creates a new operating system process for the thread.

iii) Java will throw a RuntimeException if an object's writeObject method attempts to write a non-serializable object to an ObjectOutputStream.

iv) Java's object serialization automatically compresses data before writing to improve performance.

v) If a Java class being deserialized has a different serialVersionUID than the sender's class, Java will attempt to evolve the object to the recipient's class definition.

4)

a)

i) Outline what is meant by Object Relational Mapping (ORM). **[10 marks]**

ii) Describe the three (03) challenges involved in ORM. **[15 marks]**

iii) State the use of the following JPA annotations. **[25 marks]**

a. @Entity

b. @Table

c. @Id

d. @GeneratedValue

e. @Transient

b)

i) Explain the concept of inversion of control by considering the differences between a library and a framework. **[10 marks]**

ii) Describe one (01) advantage of the use of interfaces over abstract classes in Java. **[20 marks]**

iii) A software company wants to use a hypothetical framework called "AppCore" in a desktop application that requires data persistence. The unit tests should use an in-memory database, while the production version should use a file-based database.

What are the configurations that need to be done or that will be done automatically by AppCore to facilitate this setup? **[20 marks]**

...............End of the paper...............