UNIVERSITY OF RUHUNA

Faculty of Engineering

End-Semester 7 Examination in Engineering: March 2022

Module Number: EE7210 **Module Name: Object Oriented Design Patterns and Principles**

[Three Hours]

[Answer **all questions**, each question carries 10 marks]

Q1 a) Briefly explain the following terms.

i) Single responsibility
ii) Dependency Inversion Principle

[2 Marks]

b) Answer the questions referring to the code given below.

```
import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Path;
import java.util.List;

interface Reader {
  List < String > getLines();

  String getDatabaseName();
}

class FileReader implements Reader {
  private String file = "/User/reader/records.csv";

  public List < String > getLines() {
    try {
      return Files.readAllLines(Path.of(file));
    } catch (IOException e) {
      e.printStackTrace();
      return null;
    }
  }

  @Override
  public String getDatabaseName() {
    throw new UnsupportedOperationException("No database name for the
file.");
  }
}
```

```
class RecordProcessor {
   public void run() {
      Reader reader = new FileReader();
      for (String line: reader.getLines()) {
         processLine(line);
      }
   }

   private void processLine(String line) {
      // Processing code
   }
}
```

i) What are the SOLID principles violated in the code given below?

[2 Marks]

ii) Rewrite the code to solve the problems identified in i).

[6 Marks]

Q2 a) List 2 benefits you expect by refactoring an existing working code?

[2 Marks]

b) How do the automated tests help when refactoring the code?

[1 Mark]

c) List down all the important unit tests for the "BonusCalculator" class in the code given below.
Example: return 500 If employee rank is above 5 and the score is above 80

```
class Employee {
   private int id;

   //employees are ranked from 1 to 10
   private int rank;
   private String name;
   private String designation;

   // getters and setters
}

interface ScoreCalculator {
   // score is an integer value from 0 to 100
   int getScore(int rank, String designation);
}

interface UserDataStore {
   Employee getUser();
}

class BonusCalculator {
```

```
  private ScoreCalculator scoreCalculator;

  public double calculate(Employee employee) {
    if (employee.getRank() < 5) {
      return 100;
    } else {
      int score = scoreCalculator.getScore(employee.getRank(),
employee.getDesignation());
      if (score >= 80) {
        return 500;
      } else {
        return 250;
      }
    }
  }
}
```

[3 Marks]

d) Write the actual unit test using Mockito and Junit for the the test "return 500 If employee rank is above 5 and the score is above 80"

[4 Marks]

Q3 a) What is the difference between factory and abstract factory design patterns?

[2 Marks]

b) List down the benefits of abstract factory pattern.

[2 Marks]

c) Design the class diagram for the solution based on the abstract factory design pattern for the scenario given below.

"A car manufacturer is making cars under 3 categories named Luxury, Micro and Mini. The manufacturer has 3 car factories in the UK, USA and UAE. All these countries manufacture vehicles of all three categories. The entertainment systems of each category are different in each country."

[6 Marks]

Q4 a) What is the problem solved by the "Decorator" design pattern?

[2 Marks]

b) A programmer was assigned to write a program to build a GUI creation tool. Answer the following questions, based on the requirement given below.

The GUI creation tool supports different types of GUI components such as Panel component, Text view component, Input box component, Button component etc. Three types of borders, Plain, 3D and Fancy, can be added to each GUI component. The Panel component can be associated with Horizontal and Vertical scrollbars.

i) "Decorator design pattern can be used to implement the above requirement". Provide facts to justify this statement.

[3 Marks]

ii) Draw the class diagram for the above requirement, using the decorator pattern (or any other pattern you think is suitable).

[5 Marks]

Q5 a) What is "High Cohesion" in General Responsibility Assignment Software Patterns (GRASP)?

[2 Marks]

b) Explain how high cohesion leads into low coupling using a suitable example based on the Observer pattern.

[3 Marks]

c) Draw the class diagram for the example you mentioned in part (b).

[5 Marks]