UNIVERSITY OF RUHUNA

Faculty of Engineering

End-Semester 7 Examination in Engineering: July2017

Module Number: EE7205 **Module Name: Object Oriented Design Patterns and Principles**

[3 Hours]

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

Q1 a) What is Open/Close principal?

[2.0 Marks]

b) Explain what is 'Liskov Substitution' in SOLID by using an example.

[2.0 Marks]

c) Identify the problems in the following code segment in terms of best practices. Rewrite with the fixes.

```
public class MySqlConnection {

        public static JDBCConnection connect(String connectionString){
            return new JDBCConnection(connectionString);
        }

        public static List<Object> execute(JDBCConnection connection, String sql)
{
            JDBCResult result = connection.execute(sql);
            return result.getResultSet();
        }
}
```

[4.0 Marks]

d) How does cohesion help to achieve loose coupling?

[2.0 Marks]

Q2 a) Right usage of Design Patterns has lots of benefits. List down 3 of them.

[3.0 Marks]

b) Explain "Command Pattern" buy providing a suitable example from "Java".

[3.0 Marks]

c) What are the differences and similarities between 'Interpreter' and 'Filter' patterns?

[4.0 Marks]

Q3 a) Explain the importance of Proxy pattern using a suitable example.

[4.0 Marks]

b) Draw the class diagram for the above example scenario.

[2.0 Marks]

c) What is the difference between proxy and the decorator?

[4.0 Marks]

Q4 a) Explain the "Information Expert" pattern in GRASP.

[4.0 Marks]

b) "Creator" pattern in GRASP claims, "Class B should be responsible for creating instances of class A if, Instances of B have the initializing information for instances of A and pass it on creation."

i) Explain this statement using a suitable example.

[4.0 Marks]

ii) Write down two other conditions that Class B can become the creator of Class A?

[2.0 Marks]

Q5

```
public classCachedStore implements Store{

    private Store originalStore;
    private Map cache = new HashMap();

    public CachedStore(Store originalStore){
        this.originalStore = originalStore;
}

    public void put(Object, key, Object value){
        //write to both cache and original store
        originalStore.put(key, value);
        cache.put(key, value);
    }

    public Object get(Object key) {
        if(!cache.getKeys().contains(key)) {
            Object value = originalStore.get(key);
            cache.put(key, value);
        }
return cache.get(key);
    }
}
```

a) Identify the list of unit tests you should perform on the above class.

[4.0 Marks]

b) Implement one of the happy path tests.

[4.0 Marks]

c) Implement a test to cover an Error scenario.

[2.0 Marks]