184

B.Sc. Engg. SWE 3rd Semester **8 March 2019 (Afternoon)**

ISLAMIC UNIVERSITY OF TECHNOLOGY (IUT)

ORGANISATION OF ISLAMIC COOPERATION (OIC)

Department of Computer Science and Engineering (CSE)

MID SEMESTER EXAMINATION **WINTER SEMESTER, 2018-2019**

DURATION: 1 Hour 30 Minutes **FULL MARKS: 75**

SWE 4301: Object Oriented Concepts II

Programmable calculators are not allowed. Do not write anything on the question paper.

There are **4 (four)** questions. Answer **3 (three)** of them **including question 1**.

Figures in the right margin indicate marks.

Question 1 is mandatory to answer

1. You recently got a messy project to work with and the `Ride` class in Figure 1 is part of it.

```
class Ride {
    String vt;
    int distance;
    int nop;

    int getFare() {
        int f;
        if (vt == "sedun") {
            f = (50 + distance * 30) / nop;
        } else if (vt == "motorbike") {
            f = Math.max(25, distance * 20) / nop;
        } else {
            if (distance < 10)
                f = 300 / nop;
            else
                f = distance * 30 / nop;
        }

        return f - (f % 5);
    }

    boolean isRideValid() {
        if (vt == "sedun") {
            return nop <= 4 && distance <= 25;
        } else if (vt == "sevenseater") {
            return nop <= 7 && distance >= 10;
        } else {
            return nop == 1 && distance <= 10;
        }
    }
}
```

Figure 1

a) i. Detect at least 5 unique code smells and find at least 10 lines of code with smells. Mention the line number where a particular code smell is found. 12+3

ii. Two of the code smells are worse than others, which are those?

b) The code in Figure 1 needs refactoring. Draw a class diagram or write a code with empty methods that represents the expected code after refactoring. 10

2. a) One of your friends is complaining that the `Ride` class in Figure 1 is not open for extension. 3+2+4

i. Give an example of an extension that your friend might be referring to.

ii. Find the locations of the code that requires modification to implement the extension you mentioned in answer 2a(i).

iii. What are the benefits of the design you suggested in answer 1(b) has over the original design, in terms of the extension? Note that there is a lot more code in the project, the `Ride` class is only a part of it.

b) State the fundamental concept of the following design principles. Use maximum 3 sentences for each. 3x3

i. Open Closed Principle (OCP)

ii. Liskov Substitution Principle (LSP)

iii. Dependency Inversion Principle (DIP)

c) Graph in Figure 2 represents a program. The nodes represent type/class, the filled edges represent program flow direction and the dashed edges represent source code dependency direction. This program does not follow DIP. How would the graph look like if the program did follow DIP? Note that you might require to add new nodes. 7

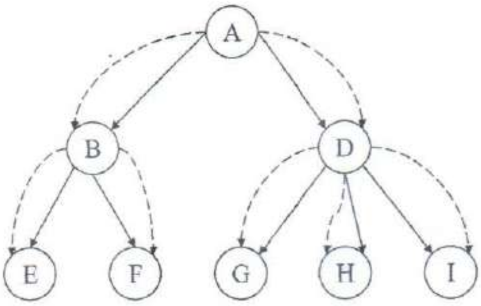

Figure 2: Dependency graph of a program

3. a) You heard your friend saying, "That critical bug was fixed as a result of refactoring". 2+5+5

i. Define refactoring.

ii. Explain why the statement above incorrectly uses the term refactoring.

iii. Based on the definition of refactoring, explain how unit testing is related to it.

b) Unit testing is difficult without dependency inversion. Explain with an example. 6

c) Give a simple code example where LSP is violated. 7

4. a) There is a popular phrase saying "prefer composition over inheritance". 2+5

i. Inheritance gives two benefits, one of which can also be achieved by composition. What is this common benefit of composition and inheritance?

ii. What is your opinion regarding this phrase? Defend your opinion.

186

b) 4+3

```
class A {
    B b;
    A() {
        b = new B();
    }
}
```

Figure 3: Program for Question 4(b)

i. Rewrite the code in Figure 3 so that it uses dependency injection.
ii. Between classes A and B, which one is dependent and which one is dependee?

c) The `SimpleList` class in Figure 4: 4 only supports `String` type. Write a generic version of this class so that any type can be supported. 7

```
class SimpleList {
    String[] allItems;

    SimpleList(int capacity) {
        allItems = new String[capacity];
    }

    void set(int index, String item) {
        allItems[index] = item;
        lastIndex++;
    }

    String get(int index) {
        return allItems[index];
    }
}
```

Figure 4: Program for Question 4(c)

d) Explain class variable and instance variable with code examples. 4