ISLAMIC UNIVERSITY OF TECHNOLOGY (IUT)

ORGANISATION OF ISLAMIC COOPERATION (OIC)

Department of Computer Science and Engineering (CSE)

SEMESTER FINAL EXAMINATION — **WINTER SEMESTER, 2021-2022**

DURATION: 3 HOURS — **FULL MARKS: 150**

SWE 4301: Object Oriented Concepts II

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

Answer all **6 (six)** questions. Marks of each question and corresponding CO and PO are written in the right margin with brackets.

---

1. a) "A good design should exhibit high cohesion and low coupling."-Justify the assertion by considering two principles of SOLID. — 8 (CO1) (PO1)

   b) Draw a diagram to illustrate how SOLID principles are related with each other. — 10 (CO1) (PO1)

   c) "Violation of Liskov Substitution Principle (LSP) is a latent violation of Open Close Principle (OCP)."- Explain the statement with an example. — 7 (CO1) (PO1)

2. a) Figure 1 represents a program as a graph. 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 Dependency Inversion Principle (DIP). — 7 (CO4) (PO1)

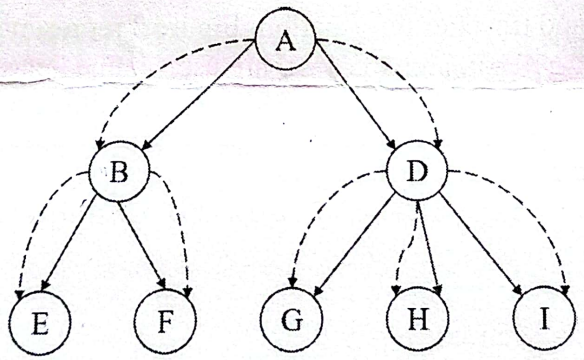

Figure 1: Dependency graph of a program

With a brief explanation, redraw the graph considering the program follows DIP.

b) In a software development team, one of the developers encouraged others to perform refactoring by saying that "Critical bug will be fixed as a result of refactoring." — 5+5 (CO1) (PO1)

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

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

c) Briefly explain each of the following code smells with example: — 4 × 2 (CO1) (PO1)

   i. Refused Bequest

   ii. Middle Man

3. a) How is encapsulation different from Information hiding? Justify your answer. — 6 (CO1) (PO1)

   b) What is clean code? Write four practices to ensure clean code. — 5 (CO1) (PO1)

   c) Briefly describe Retention Policy of custom `Annotation`. — 5 (CO1)

d) The SimpleQueue class in Figure 2 presents a queue implementation. 5+4 (CO3) (PO1)

```
1  class SimpleQueue {
2      Queue<Integer> queue;
3      SimpleQueue() {
4          this.queue = new LinkedList<>();
5      }
6      void enqueue(Integer val){
7          this.queue.add(val);
8      }
9      Integer dequeue(){
10         Integer value = this.queue.remove();
11         return value;
12     }
13 }
```

Figure 2: Code Snippet for Question 3.d)

i. Figure 2 only supports Integer data type. Write a generic version of this class so that any type of Comparable object can be supported.

ii. Write two advantages and disadvantages of Generics.

4. a) AmazeSoft wants to build a program that can generate tickets for different vehicles like buses, airplanes with traveller information, for example, name, age, and gender. The program can also print a full-color or grayscale ticket. A ticket has other details like fare price, source, destination, departure time, arrival time, and traveller information. Figure 3 represents the initial code to fulfill those requirements. `Traveller`, and `Ticket` are user-defined classes, and other methods are implemented as well. You can assume that the code has no compile time error.

```
1  class TicketGeneratorAndPrinter {
2      // remember to reset these flags to change vehicle and
          color printing
3      int vflag = 1;
4      boolean cprint = true;
5
6      void generateTicket(Traveller traveller){
7          Ticket ticket;
8          if(vflag==1) ticket = busTicket(traveller);
9          else ticket = airTicket(traveller);
10     }
11
12     void printTicket(Ticket ticket){
13         if(cprint) fullcolorPrint(ticket);
14         else grayscalePrint(ticket);
15     }
16 }
```

Figure 3: Code for Question 4.a)

i. Figure 3 has several design smells. With a brief explanation, identify at least 4 design smells. Mention the line numbers. 8 (CO4) (PO1)

ii. Rewrite the code satisfying SOLID principles to remove design smells. You can use Class Diagram or any preferred Object Oriented Programming language. 12 (CO4) (PO2)

iii. Write a lambda expression that will print all tickets which has the same destination address. 5 (CO3) (PO1)

5. a) The code snippet in Figure 4 contains code smells.

```
1  public class Account {
2      String type;
3      String accountNumber;
4      private int amount;
5
6      public void debit(int debit) throws Exception{
7          if(amount <= 500)
8            throw new Exception("Minimum must be > 500");
9          amount = amount-debit;
10         System.out.println("Now amount is " + amount);
11     }
12
13     public void transfer(Account from, Account to, int
         creditAmount) throws Exception{
14         if(from.amount <= 500)
15           throw new Exception("Minimum must be > 500");
16         to.amount = amount+creditAmount;
17     }
18 }
```

Figure 4: Code Snippet for Question 5. a)

i. Identify two code smells and rewrite the code after refactoring. 6 (CO4) (PO1)

ii. Using Reflection, write a method that will check whether the `amount` field is `private` or not. 5 (CO3) (PO1)

iii. Write a custom exception named as `BalanceException` and rewrite the `debit` method to throw `BalanceException` instead of `Exception`. 6 (CO3) (PO1)

iv. Write a test case that will validate minimum amount for `transfer` method. 3 (CO2) (PO1)

b) Differentiate Mutable and Immutable Object with an example. 5 (CO1) (PO3)

6. a) Describe the use of the `final` keyword. Write the difference between two declarations in Figure 5. 5 (CO1) (PO1)

```
1 final double pi = 3.1416;
2 final Account[] accounts = new Account[5];
```

Figure 5: Code Snippet for Question 6.a)

b) How are accessors and mutators related with pure and impure functions? 5 (CO1) (PO1)

c) What is Mutual Exclusion? How can mutual exclusion be achieved in Java? 5 (CO1) (PO1)

d) Demonstrate Producer/Consumer Relationship in Multi-Threading without synchronization by using your preferred programming language. 10 (CO1) (PO1)