# University of Information Technology & Sciences (UITS)
## Faculty of Science and Engineering
## Department of Computer Science and Engineering
### Program: B.Sc. in CSE
### Term Final Examination, Spring 2025
### Course Title: Object-Oriented Programming Language
### Course Code: CSE0613121

**Marks: 50** **Time: 3*(three)* hours**

**(Answer all questions.)**

Q.No. **Marks**

1. **a)** Analyze the following program and write down the exact output it generates upon execution. [04]

```
public class Demo {
  public static void main(String[] args) {
    String first = "Rajiv", last = "Ahmed";
    String full = first.concat(last);

    System.out.println("Full Name: " + full);
    System.out.println("Length: " + full.length());
    System.out.println("Char at 3: " + full.charAt(3));
    System.out.println("Index of 'a': " + full.indexOf('a'));
  }
}
```

**b)** You are given a task to design a Calculator class that encapsulates two integer values and performs multiplication using a private method. Follow the instructions below to complete the task: Implement the class named Calculator with the following specifications: [06]

1. Declare two private integer variables, a and b.
2. Define a private method named multiply() that returns the product of a and b.
3. Create a public method named multiplyNumbers() that calls the private multiply() method and returns the result.

Develop a Java program that sets and retrieves private variables using Java features, invokes the multiplyNumbers() method, and displays the result of the multiplication.

2. **a)** The following code demonstrates the use of abstract classes [04]

```
abstract class Device {
  abstract void start();
}

abstract class Mobile extends Device {
  void start() {
    System.out.println("Mobile started");
```

```
} }
```

Here, the Mobile class overrides all abstract methods from the Device class. Can you remove the abstract keyword from Mobile? If yes rewrite the code and demonstrate by creating of it in a main() method to call the start() method.

b) You are developing a program for a smart home automation system. Different devices (like lights and fans) can be switched on or off, and each should display a custom message when turned on. However, for turning off, all devices should follow a common behavior display "Device turned off". [06]

To achieve this:

- Create an interface SmartDevice with two methods:
  - void turnOn(); – an abstract method (each device has its own implementation)
  - default void turnOff() – a default method that prints "Device turned off".

Then:

1. Create two classes: Light and Fan that implement the SmartDevice interface.
2. In each class, override the turnOn() method to print:
   - "Light is now ON" (in Light)
   - "Fan is now ON" (in Fan)
3. In a Main class, create objects of both classes and call turnOn() and turnOff().

3. a) Show the output of the following program, if no output explains the issue and solution. [03]

```
public class Test {
    final int a = 0;
    final int b = 1;
    final int c = 2;

    public static void main(String[] args) {
        Test obj = new Test();
        obj.a = 10;
        obj.b = 20;
        obj.c = 30;
        System.out.println("a = " + a);
        System.out.println("b = " + b);
        System.out.println("c = " + c);
    }
}
```

b) Create a base class called Shape with a method void draw() that prints "Drawing Shape". [04]

- Create two classes Circle and Rectangle that extend the Shape class.
- Override the draw() method in both Circle and Rectangle classes to print "Drawing Circle" and "Drawing Rectangle" respectively.

Write a Main class with the main method to:

- Create objects of Circle and Rectangle.
- Call the draw() method on both objects.

c) Differentiate between method overloading and method overriding in Java. [03]

4. a) There are a java code which is performing simple mathematical operation: [04]

```
class MarksCalculator {
    public static void main(String[] args) {
        int[] marks = {85, 90, 88};
        int index = 5;
        int divisor = 0;
        int result = marks[index] / divisor;
        System.out.println("Result of division: " + result);
    }}
```

Modify the above code to:

1. Handle ArithmeticException when dividing by zero.
2. Handle ArrayIndexOutOfBoundsException when the user enters an invalid index.
3. Use a try-catch block to display appropriate error messages like:
    - "Error: Cannot divide by zero."
    - "Error: Invalid index entered."

b) Explain the difference between an Error and an Exception in Java. In your answer, mention their causes, whether they can be handled, and give examples of each. [02]

c) Consider the following Figure:01 [04]

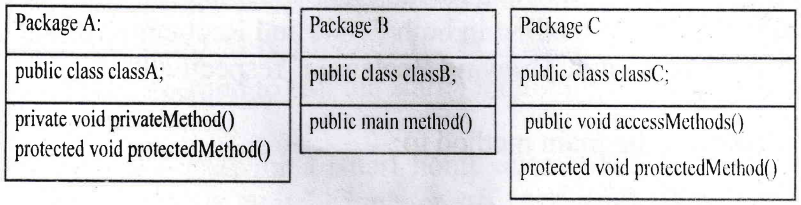


Figure :01

In classC, the method accessMethods() tries to call protectedMethod() from classA. Explain why this is allowed even though classA and classC are in different packages. Design a java code based on this feature.

5. a) Create a Class "Operations", with three methods: [03]

i. int SUB (int a, int b)

ii. int SUB (int a, int b, int c)

iii. double SUB (double a, int b)

b) Explain the life cycle/states of a thread in Java. [03]

c Write a Java program using multithreading where two threads increment a shared counter and print messages in the following format: [04]

Thread-0: 1

Thread-1: 1

Thread-0: 2

Thread-1: 2

Thread-0: 3

Thread-1: 3