**MODULE – 1**

**DESIGN PATTERNS AND PRINCIPLES**

**Exercise 1: Implementing the Singleton Pattern**

**CODE:**

**Logger.java**

public class Logger {

    private static Logger instance;

    private Logger() {

        System.out.println("Logger Initialized.");

    }

    public static Logger getInstance() {

        if (instance == null) {

            instance = new Logger();

        }

        return instance;

    }

    public void log(String message) {

        System.out.println("Log: " + message);

    }

}

**LoggerTest.java**

public class LoggerTest {

    public static void main(String[] args) {

        Logger logger1 = Logger.getInstance();

        Logger logger2 = Logger.getInstance();

        logger1.log("This is the first log message.");

        logger2.log("This is the second log message.");

        if (logger1 == logger2) {

            System.out.println("Both logger1 and logger2 are the same instance.");

        } else {

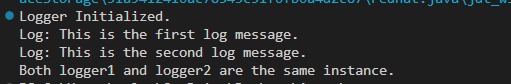
            System.out.println("Different instances exist.");

        }

    }

}

**OUTPUT :**

****

**Exercise 2: Implementing the Factory Method Pattern**

**CODE:**

**Document.java**

public interface Document {

    void open();

}

**DocumentFactory.java**

public abstract class DocumentFactory {

    public abstract Document createDocument();

}

**DocumentFactoryTest.java**

public class DocumentFactoryTest {

    public static void main(String[] args) {

        DocumentFactory wordFactory = new WordDocumentFactory();

        Document word = wordFactory.createDocument();

        word.open();

        DocumentFactory pdfFactory = new PdfDocumentFactory();

        Document pdf = pdfFactory.createDocument();

        pdf.open();

        DocumentFactory excelFactory = new ExcelDocumentFactory();

        Document excel = excelFactory.createDocument();

        excel.open();

    }

}

**ExcelDocument.java**

public class ExcelDocument implements Document {

    public void open() {

        System.out.println("Opening an Excel document.");

    }

}

**ExcelDocumentFactory.java**

public class ExcelDocumentFactory extends DocumentFactory {

    public Document createDocument() {

        return new ExcelDocument();

    }

}

**PdfDocument.java**

public class PdfDocument implements Document {

    public void open() {

        System.out.println("Opening a PDF document.");

    }

}

**PdfDocumentFactory.java**

public class PdfDocumentFactory extends DocumentFactory {

    public Document createDocument() {

        return new PdfDocument();

    }

}

**WordDocument.java**

public class WordDocument implements Document {

    public void open() {

        System.out.println("Opening a Word document.");

    }

}

**WordDocumentFactory.java**

public class WordDocumentFactory extends DocumentFactory {

    public Document createDocument() {

        return new WordDocument();

    }

}

**OUTPUT :**

****