**Week 1**

**Design principles & Patterns**

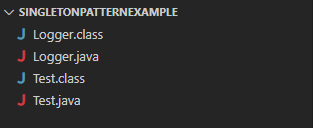
**Exercise 1: Implementing the Singleton Pattern**

**Scenario:**

You need to ensure that a logging utility class in your application has only one instance throughout the application lifecycle to ensure consistent logging.

Answer:

FOLDER STRUCTURE



CODE

Logger.java

public class Logger {

    private static Logger instance = new Logger();

    private Logger() {

        System.out.println("Initialization of Logger");

    }

    public static Logger getInstance(){

        return instance;

    }

    public void log(String message){

        System.out.println("[LOG]: " + message);

    }

}

Test.java

public class Test {

    public static void main(String[] args){

        Logger logger1 = Logger.getInstance();

        Logger logger2 = Logger.getInstance();

        logger1.log("Start...log 1");

        logger2.log("Start...log 2");

        if (logger1 == logger2) {

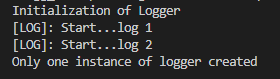
            System.out.println("Only one instance of logger created");

        }

    }

}

OUTPUT



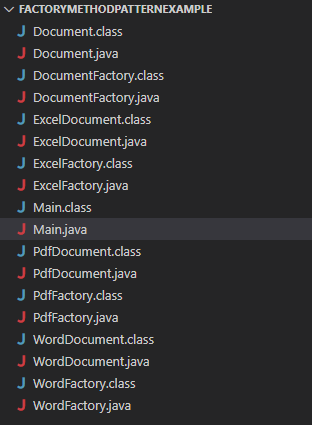
**Exercise 2: Implementing the Factory Method Pattern**

**Scenario:**

You are developing a document management system that needs to create different types of documents (e.g., Word, PDF, Excel). Use the Factory Method Pattern to achieve this.

Answer:

FOLDER STRUCTURE



CODE

Document.java

public interface Document {

    void open();

}

WordDocument.java

public class WordDocument implements Document{

    public void open(){

        System.out.println("Creating word document");

    }

}

PdfDocument.java

public class PdfDocument implements Document {

    public void open() {

        System.out.println("Creating pdf document");

    }

}

ExcelDocument.java

public class ExcelDocument implements Document {

    public void open() {

        System.out.println("Creating excel document");

    }

}

DocumentFactory.java

public abstract class DocumentFactory {

    public abstract Document createDocument();

}

WordFactory.java

public class WordFactory extends DocumentFactory {

    public Document createDocument() {

        return new WordDocument();

    }

}

PdfFactory.java

public class PdfFactory extends DocumentFactory {

    public Document createDocument() {

        return new PdfDocument();

    }

}

ExcelFactory.java

public class ExcelFactory extends DocumentFactory {

    public Document createDocument() {

        return new ExcelDocument();

    }

}

Main.java

public class Main {

    public static void main(String[] args) {

        DocumentFactory wordFactory = new WordFactory();

        DocumentFactory pdfFactory = new PdfFactory();

        DocumentFactory excelFactory = new ExcelFactory();

        Document wordDoc = wordFactory.createDocument();

        Document pdfDoc = pdfFactory.createDocument();

        Document excelDoc = excelFactory.createDocument();

        wordDoc.open();

        pdfDoc.open();

        excelDoc.open();

    }

}

OUTPUT

