**Week 1-Design Patterns and Principles**

**Exercise 1: Implementing the Singleton Pattern**

**Logger.java**

public class Logger {

// Step 1: Private static instance of Logger

private static Logger instance;

// Step 2: Private constructor to prevent instantiation

private Logger() {

System.out.println("Logger instance created.");

}

// Step 3: Public method to get the single instance

public static Logger getInstance() {

if (instance == null) {

instance = new Logger(); // created only once

}

return instance;

}

// Example log method

public void log(String message) {

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

}

}

**Main.java**

public class Main {

public static void main(String[] args) {

// Get Logger instance multiple times

Logger logger1 = Logger.getInstance();

Logger logger2 = Logger.getInstance();

// Log messages

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

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

// Verify both instances are same

if (logger1 == logger2) {

System.out.println("Only one Logger instance is used.");

} else {

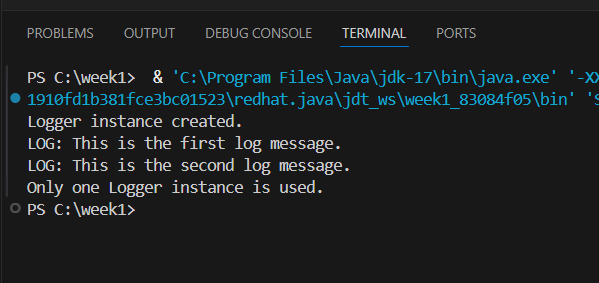
System.out.println("Different Logger instances exist (Error!)");

}

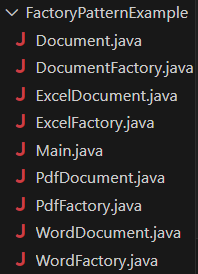
}

}

**OUTPUT**



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

****

**Document.java**

package FactoryPatternExample;

public interface Document {

    void open();

}

**DocumentFactory.java**

package FactoryPatternExample;

public abstract class DocumentFactory {

    public abstract Document createDocument();

}

**ExcelDocument.java**

package FactoryPatternExample;

public class ExcelDocument implements Document {

    public void open() {

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

    }

}

**ExcelFactory.java**

package FactoryPatternExample;

public class ExcelFactory extends DocumentFactory {

    public Document createDocument() {

        return new ExcelDocument();

    }

}

**PdfDocument**

package FactoryPatternExample;

public class PdfDocument implements Document {

    public void open() {

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

    }

}

**PdfFactory**

package FactoryPatternExample;

public class PdfFactory extends DocumentFactory {

    public Document createDocument() {

        return new PdfDocument();

    }

}

**WordDocument**

package FactoryPatternExample;

public class WordDocument implements Document {

    public void open() {

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

    }

}

**WordFactory**

package FactoryPatternExample;

public class WordFactory extends DocumentFactory {

    public Document createDocument() {

        return new WordDocument();

    }}

**Main.java**

package FactoryPatternExample;

public class Main {

    public static void main(String[] args) {

        // Create a Word document using factory

        DocumentFactory wordFactory = new WordFactory();

        Document word = wordFactory.createDocument();

        word.open();

        // Create a PDF document using factory

        DocumentFactory pdfFactory = new PdfFactory();

        Document pdf = pdfFactory.createDocument();

        pdf.open();

        // Create an Excel document using factory

        DocumentFactory excelFactory = new ExcelFactory();

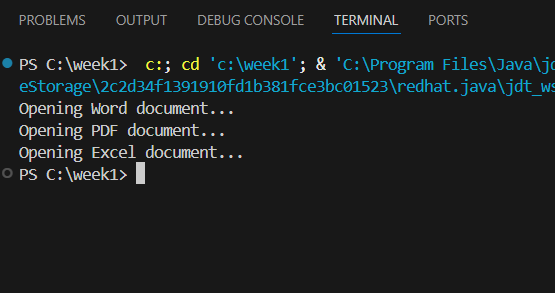
        Document excel = excelFactory.createDocument();

        excel.open();

    }

}

**Output**

d