**Week 1-Mandatory Hands on:**

**Exercise 1: Implementing the Singleton Pattern**

**Logger.java**

package Harsha;

public class Logger {

private static Logger *instance*;

private Logger() {

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

}

public static Logger getInstance() {

if (*instance* == null) {

*instance* = new Logger();

}

return *instance*;

}

public void log(String message) {

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

}

}

**Main.java**

package Harsha;

public class Main {

public static void main(String[] args) {

Logger logger1 = Logger.*getInstance*();

logger1.log("First message");

Logger logger2 = Logger.*getInstance*();

logger2.log("Second message");

if (logger1 == logger2) {

System.*out*.println("Both logger instances are the same. Singleton verified.");

} else {

System.*out*.println("Logger instances are different. Singleton failed.");

}

}

}

**OUTPUT:**

**A screenshot of a computer

AI-generated content may be incorrect.**

**Exercise 2: Implementing Factory Method**

**FactoryMethodExample.java**

package Harsha;

public class FactoryMethodExample {

public static void main(String[] args) {

DocumentFactory wordFactory = new WordDocumentFactory();

Document wordDoc = wordFactory.createDocument();

wordDoc.open();

DocumentFactory pdfFactory = new PdfDocumentFactory();

Document pdfDoc = pdfFactory.createDocument();

pdfDoc.open();

DocumentFactory excelFactory = new ExcelDocumentFactory();

Document excelDoc = excelFactory.createDocument();

excelDoc.open();

}

}

interface Document {

void open();

}

class WordDocument implements Document {

@Override

public void open() {

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

}

}

class PdfDocument implements Document {

@Override

public void open() {

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

}

}

class ExcelDocument implements Document {

@Override

public void open() {

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

}

}

abstract class DocumentFactory {

public abstract Document createDocument();

}

class WordDocumentFactory extends DocumentFactory {

@Override

public Document createDocument() {

return new WordDocument();

}

}

class PdfDocumentFactory extends DocumentFactory {

@Override

public Document createDocument() {

return new PdfDocument();

}

}

class ExcelDocumentFactory extends DocumentFactory {

@Override

public Document createDocument() {

return new ExcelDocument();

}

}

**OUTPUT:**

**A screenshot of a computer

AI-generated content may be incorrect.**