**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.**

Logger.java

package SingletonPattern;

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);

}

}

Test.java

package SingletonPattern;

public class Test {

public static void main(String[] args) {

Logger logger1 = Logger.getInstance();

Logger logger2 = Logger.getInstance();

System.out.println("Same instance? " + (logger1 == logger2));

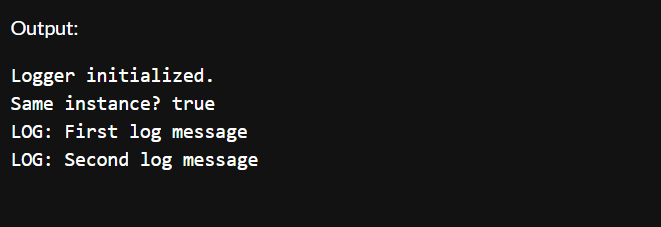
logger1.log("First log message");

logger2.log("Second log message");

Logger logger3 = Logger.getInstance();

}

}



**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.**

Document.java

package FactoryMethod;

interface Document {

void open();

}

WordDocument.java

package FactoryMethod;

class WordDocument implements Document {

@Override

public void open() {

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

}

}

PdfDocument.java

package FactoryMethod;

class PdfDocument implements Document {

@Override

public void open() {

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

}

}

ExcelDocument.java

package FactoryMethod;

class ExcelDocument implements Document {

@Override

public void open() {

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

}

}

DocumentFactory.java

package FactoryMethod;

abstract class DocumentFactory {

public abstract Document createDocument();

}

WordDocumentFactory.java

package FactoryMethod;

class WordDocumentFactory extends DocumentFactory {

@Override

public Document createDocument() {

return new WordDocument();

}

}

PdfDocumentFactory.java

package FactoryMethod;

class PdfDocumentFactory extends DocumentFactory {

@Override

public Document createDocument() {

return new PdfDocument();

}

}

ExcelDocumentFactory.java

package FactoryMethod;

class ExcelDocumentFactory extends DocumentFactory {

@Override

public Document createDocument() {

return new ExcelDocument();

}

}

FactoryTest.java

package FactoryMethod;

public class FactoryTest{

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();

}

}

