**Exercise 1: Implementing the Singleton Pattern**

**Solution:**

Project Name: SingletonPatternExample

Filename**:** Main.java

File Structure:

SingletonPatternExample

└── Main.java

Code:

**// Step 4: Test the Singleton Implementation**

public class Main {

public static void main(String[] args) {

// Get instances of Logger

Logger logger1 = Logger.getInstance();

Logger logger2 = Logger.getInstance();

// Use the logger

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

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

// Verify if both instances are the same

if (logger1 == logger2) {

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

} else {

System.out.println("Different instances (Singleton failed).");

}

}

}

**// Step 2 & 3: Define and Implement the Singleton Class**

class Logger {

// Create a class named Logger that has a private static instance of itself

private static Logger instance;

// Ensure the constructor of Logger is private

private Logger() {

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

}

// Provide a public static method to get the instance of the Logger class

public static Logger getInstance() {

// Ensure that only one instance is created (Singleton pattern)

if (instance == null) {

instance = new Logger();

}

return instance;

}

// Sample logging method

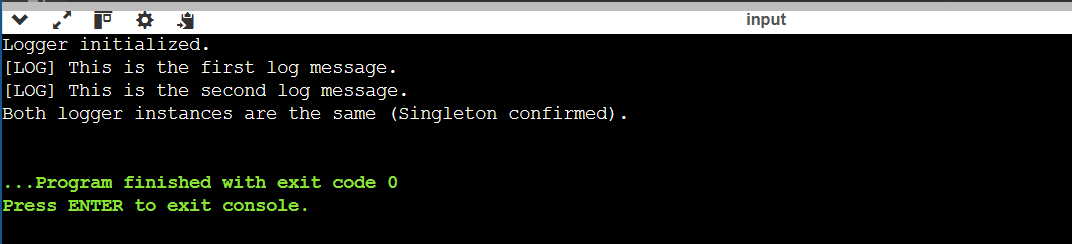
public void log(String message) {

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

}

}

Output:



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

**Solution:**

Project Name: FactoryMethodPatternExample

Filename**:** Main.java

File Structure:

FactoryMethodPatternExample

└── Main.java

Code:

**// Step 5: Test class to demonstrate the Factory Method**

public class Main {

public static void main(String[] args) {

// Creating PDF document using factory

DocumentFactory pdfFactory = new PdfDocumentFactory();

Document pdf = pdfFactory.createDocument();

pdf.open();

// Creating Word document using factory

DocumentFactory wordFactory = new WordDocumentFactory();

Document word = wordFactory.createDocument();

word.open();

// Creating Excel document using factory

DocumentFactory excelFactory = new ExcelDocumentFactory();

Document excel = excelFactory.createDocument();

excel.open();

}

}

**// Step 2: Define an interface or abstract class for document types**

interface Document {

void open();

}

**// Step 3: Create concrete document classes**

class WordDocument implements Document {

public void open() {

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

}

}

class PdfDocument implements Document {

public void open() {

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

}

}

class ExcelDocument implements Document {

public void open() {

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

}

}

**// Step 4: Implement the Factory Method**

**// Abstract factory class**

abstract class DocumentFactory {

public abstract Document createDocument();

}

// Concrete factory classes

class WordDocumentFactory extends DocumentFactory {

public Document createDocument() {

return new WordDocument();

}

}

class PdfDocumentFactory extends DocumentFactory {

public Document createDocument() {

return new PdfDocument();

}

}

class ExcelDocumentFactory extends DocumentFactory {

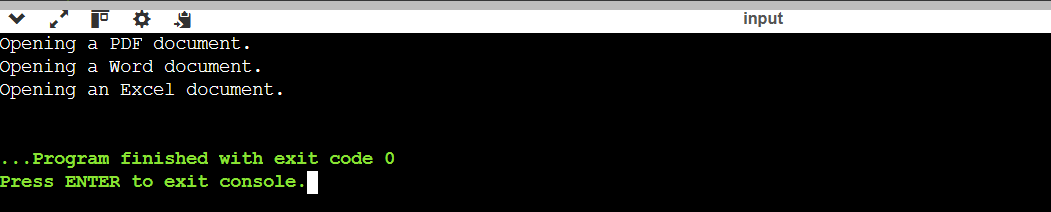
public Document createDocument() {

return new ExcelDocument();

}

}

**Output:**

****