# **Exercise 1: Implementing the Singleton Pattern**

Logger.java

package singletonpattern;

//Logger.java

public class Logger {

//Private static instance of Logger

private static Logger *instance*;

//Private constructor to prevent instantiation

private Logger() {

System.***out***.println("Logger Initialized");

}

//Public static method to provide access to the single instance

public static Logger getInstance() {

if (*instance* == null) {

*instance* = new Logger();

}

return *instance*;

}

public void log(String message) {

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

}

}

TestLogger.java

package singletonpattern;

public class TestLogger {

public static void main(String[] args) {

// Trying to get multiple instances

Logger logger1 = Logger.*getInstance*();

Logger logger2 = Logger.*getInstance*();

// Logging some messages

logger1.log("First log message.");

logger2.log("Second log message.");

// Checking if both instances are same

if (logger1 == logger2) {

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

} else {

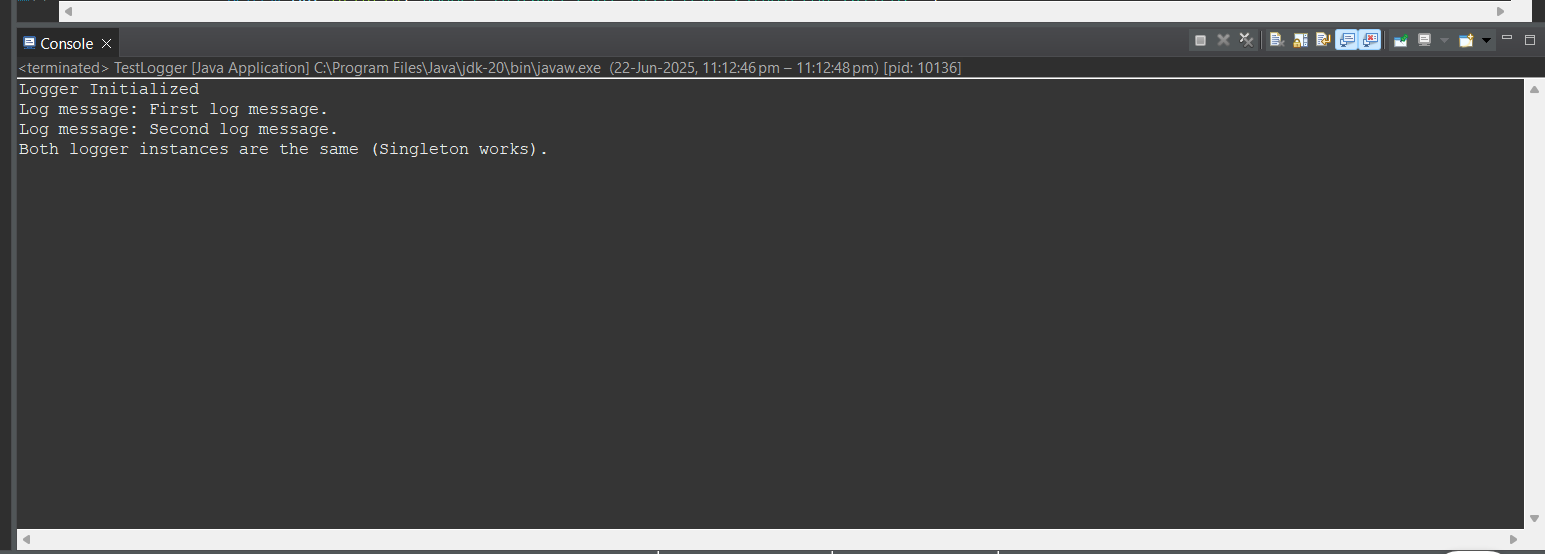
System.***out***.println("Logger instances are different (Singleton failed).");

}

}

}

Output:



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

Document.java

package factorymethodpattern;

public interface Document {

void open();

}

WordDocument.java

package factorymethodpattern;

public class WordDocument implements Document {

*@Override*

public void open() {

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

}

}

PdfDocument.java

package factorymethodpattern;

public class PdfDocument implements Document {

*@Override*

public void open() {

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

}

}

ExcelDocument.java

package factorymethodpattern;

public class ExcelDocument implements Document {

*@Override*

public void open() {

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

}

}

DocumentFactory.java

package factorymethodpattern;

public abstract class DocumentFactory {

public abstract Document createDocument();

}

WordDocumentFactory.java

package factorymethodpattern;

public class WordDocumentFactory extends DocumentFactory {

*@Override*

public Document createDocument() {

return new WordDocument();

}

}

PdfDocumentFactory.java

package factorymethodpattern;

public class PdfDocumentFactory extends DocumentFactory {

*@Override*

public Document createDocument() {

return new PdfDocument();

}

}

ExcelDocumentFactory.java

package factorymethodpattern;

public class ExcelDocumentFactory extends DocumentFactory {

*@Override*

public Document createDocument() {

return new ExcelDocument();

}

}

TestFactory.java

package factorymethodpattern;

public class TestFactory {

public static void main(String[] args) {

// Creating Word document using factory

DocumentFactory wordFactory = new WordDocumentFactory();

Document wordDoc = wordFactory.createDocument();

wordDoc.open();

// Creating PDF document using factory

DocumentFactory pdfFactory = new PdfDocumentFactory();

Document pdfDoc = pdfFactory.createDocument();

pdfDoc.open();

// Creating Excel document using factory

DocumentFactory excelFactory = new ExcelDocumentFactory();

Document excelDoc = excelFactory.createDocument();

excelDoc.open();

}

}

Output:

