**Exercise 1: Implementing the Singleton Pattern**

**Objective**:  
To ensure that only one instance of a Logger class exists throughout the application, maintaining consistent logging behavior.

Logger.java:

package cognizant\_project;

public 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);

}

}

Main.java:

package cognizant\_project;

public class Main {

public static void main(String[] args) {

// **TODO** Auto-generated method stub

Logger logger1 = Logger.*getInstance*();

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

Logger logger2 = Logger.*getInstance*();

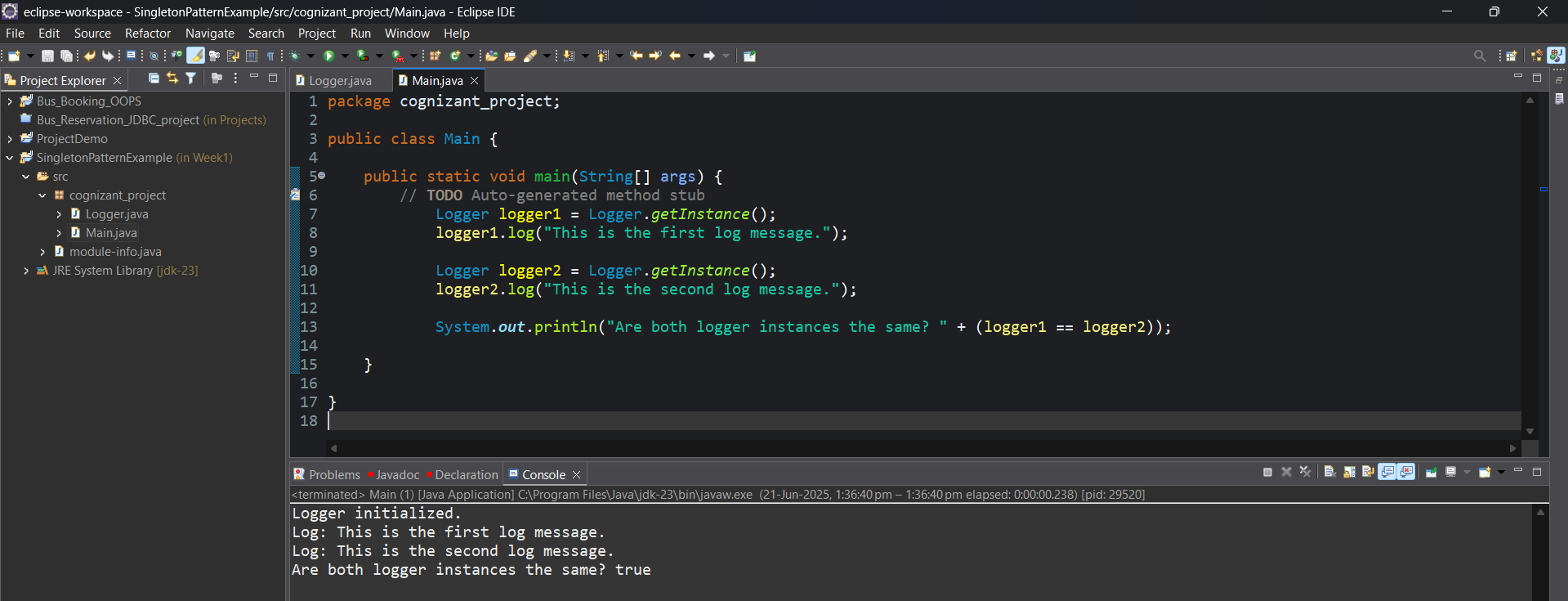
logger2.log("This is the second log message.");

System.***out***.println("Are both logger instances the same? " + (logger1 == logger2));

}

}

Output:



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

**Objective**:  
To create different types of documents (Word, PDF, Excel) using the Factory Method Pattern, which allows easy scalability and maintenance.

package Cognizant\_project;

public class Main {

// Common interface

interface Document {

void open();

}

// Concrete Document Classes

static class WordDocument implements Document {

public void open() {

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

}

}

static class PdfDocument implements Document {

public void open() {

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

}

}

static class ExcelDocument implements Document {

public void open() {

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

}

}

// Abstract Factory

static abstract class DocumentFactory {

public abstract Document createDocument();

}

// Concrete Factories

static class WordDocumentFactory extends DocumentFactory {

public Document createDocument() {

return new WordDocument();

}

}

static class PdfDocumentFactory extends DocumentFactory {

public Document createDocument() {

return new PdfDocument();

}

}

static class ExcelDocumentFactory extends DocumentFactory {

public Document createDocument() {

return new ExcelDocument();

}

}

// Main method to test factory pattern

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();

}

}

Output:  
