**Digital Nurture 4.0 Java FSE**

**Week 1 Design Patterns and Principles Handson**

**(Mandatory)**

**Exercise 01: Implementing the Singleton Pattern**

**Code:**

public class SingletonPattern {

static 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);

}

}

public static void main(String[] args) {

Logger logger1 = Logger.getInstance();

logger1.log("Logging from first logger instance.");

Logger logger2 = Logger.getInstance();

logger2.log("Logging from second logger instance.");

if (logger1 == logger2) {

System.out.println("Singleton confirmed: Both references point to the same instance.");

} else {

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

}

}

}

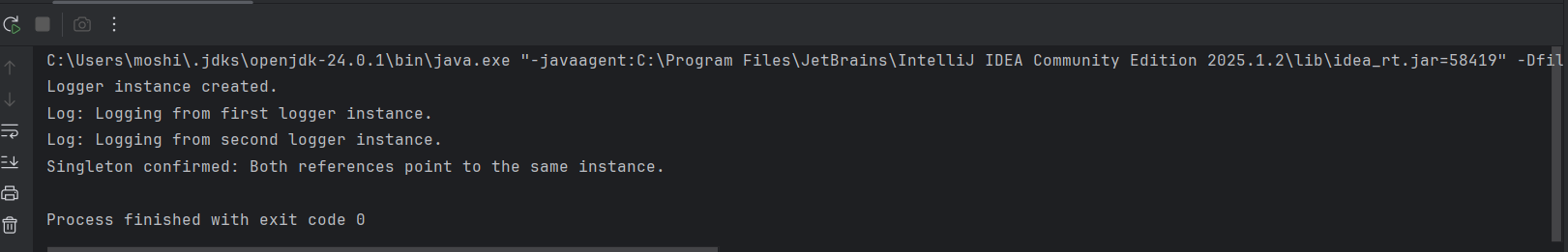
**Output:**

Logger instance created.

Log: Logging from first logger instance.

Log: Logging from second logger instance.

Singleton confirmed: Both references point to the same instance.



**Exercise 02: Implementing the Factory Method Pattern**

**Code:**

public class FactoryPattern {

interface Document {

void create();

void open();

void print();

}

static class WordDocument implements Document {

public void create() {

System.out.println("Creating Word Document");

}

public void open() {

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

}

public void print() {

System.out.println("Printing Word Document");

}

}

static class PdfDocument implements Document {

public void create() {

System.out.println("Creating PDF Document");

}

public void open() {

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

}

public void print() {

System.out.println("Printing PDF Document");

}

}

static class ExcelDocument implements Document {

public void create() {

System.out.println("Creating Excel Document");

}

public void open() {

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

}

public void print() {

System.out.println("Printing Excel Document");

}

}

static abstract class DocumentFactory {

public abstract Document createDocument();

}

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();

}

}

public static void main(String[] args) {

DocumentFactory wordFactory = new WordDocumentFactory();

Document word = wordFactory.createDocument();

word.create();

word.open();

word.print();

System.out.println();

DocumentFactory pdfFactory = new PdfDocumentFactory();

Document pdf = pdfFactory.createDocument();

pdf.create();

pdf.open();

pdf.print();

System.out.println();

DocumentFactory excelFactory = new ExcelDocumentFactory();

Document excel = excelFactory.createDocument();

excel.create();

excel.open();

excel.print();

}

}

**Output:**

Creating Word Document

Opening Word Document

Printing Word Document

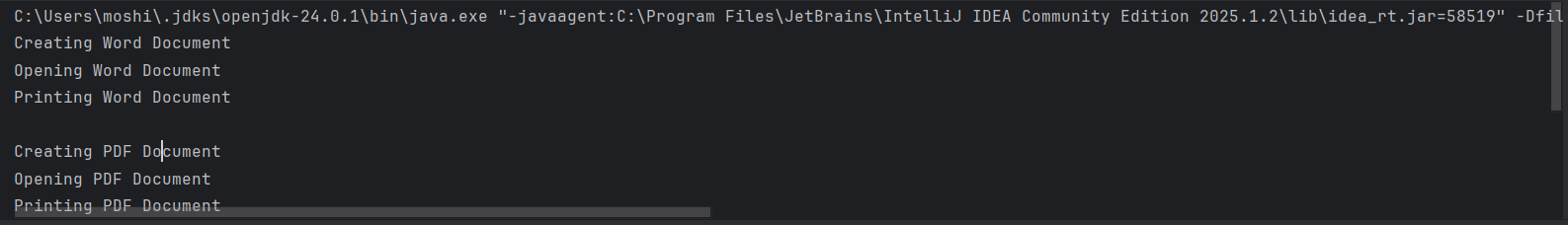
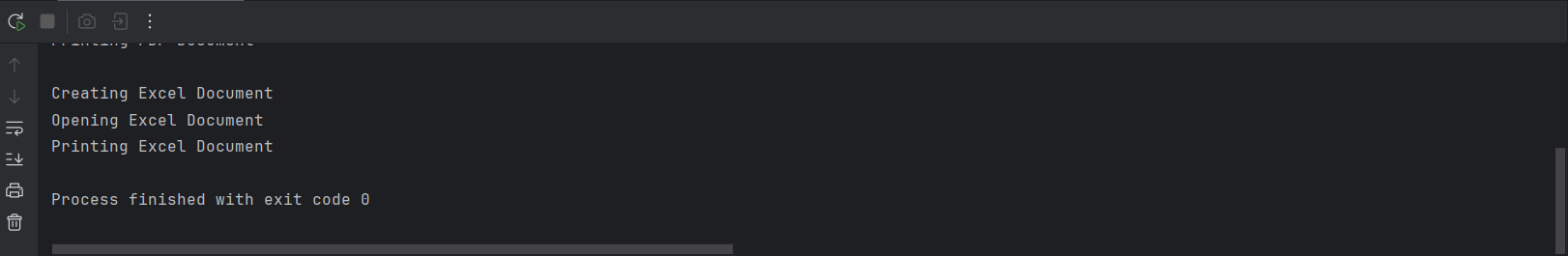
Creating PDF Document

Opening PDF Document

Printing PDF Document

Creating Excel Document

Opening Excel Document

Printing Excel Document