**Week-1**

**Design Patterns and Principles**

**Exercise 1: Implementing the Singleton Pattern**

**Problem Statement:**

Building logging utility class in an application has only one instance throughout the application lifecycle to ensure consistent logging.

**Logger.java**

**package** com.logger;

**public** **class** Logger {

**private** **static** Logger *instance*;

**private** Logger() {

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

}

**public** **static** Logger getInstance() {

System.***out***.println("Requesting Logger instance...");

**if** (*instance* == **null**) {

System.***out***.println("No instance found. Creating new Logger instance.");

*instance* = **new** Logger();

} **else** {

System.***out***.println("Logger instance already exists. Reusing it.");

}

**return** *instance*;

}

**public** **void** log(String message) {

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

}

}

**Main.java**

**package com.logger;**

**import com.logger.Logger;**

**public class Main {**

**public static void main(String[] args) {**

**Logger logger1 = Logger.*getInstance*();**

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

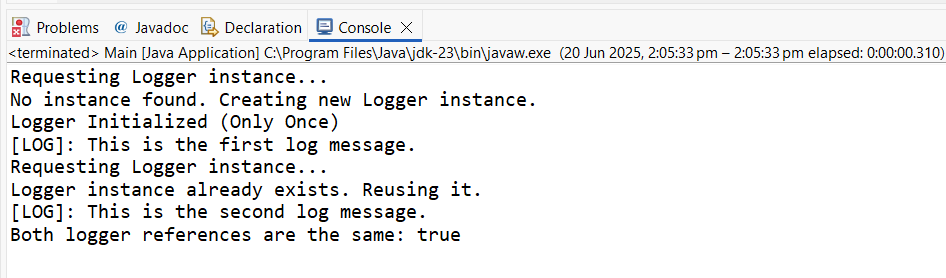
**Logger logger2 = Logger.*getInstance*();**

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

**System.*out*.println("Both logger references are the same: " + (logger1 == logger2));**

**}**

**}**

**OUTPUT:**

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

**Problem Statement:**

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** com.factory;

**public** **interface** Document {

**void** open();

}

**DocumentTypes.java**

package com.factory;

class WordDocument implements Document {

@Override

public void open() {

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

}

}

class PdfDocument implements Document {

@Override

public void open() {

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

}

}

class ExcelDocument implements Document {

@Override

public void open() {

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

}

}

**DocumentFactory.java**

package com.factory;

public abstract class DocumentFactory {

public abstract Document createDocument();

}

class WordDocumentFactory extends DocumentFactory {

@Override

public Document createDocument() {

return new WordDocument();

}

}

class PdfDocumentFactory extends DocumentFactory {

@Override

public Document createDocument() {

return new PdfDocument();

}

}

class ExcelDocumentFactory extends DocumentFactory {

@Override

public Document createDocument() {

return new ExcelDocument();

}

}

**Main.java**

package com.factory;

public class Main {

public static void main(String[] args) {

DocumentFactory wordFactory = new WordDocumentFactory();

Document word = wordFactory.createDocument();

word.open();

DocumentFactory pdfFactory = new PdfDocumentFactory();

Document pdf = pdfFactory.createDocument();

pdf.open();

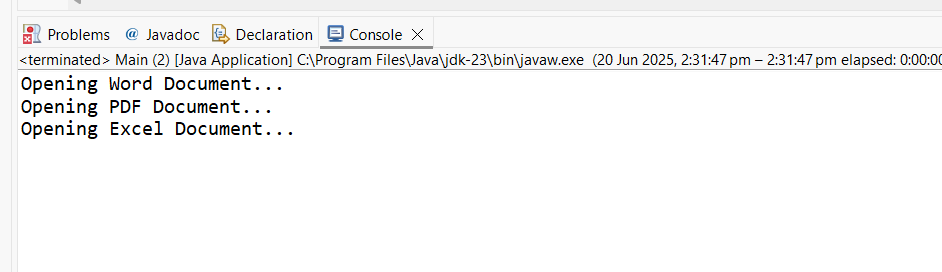
DocumentFactory excelFactory = new ExcelDocumentFactory();

Document excel = excelFactory.createDocument();

excel.open();

}

}

**OUTPUT**