**Week-1 Design Patterns and Principles**

1. **Implementing the Singleton Pattern**

**Scenario:**

You need to ensure that a logging utility class in your application has only one instance throughout the application lifecycle to ensure consistent logging.

**Code:**

**Logger.java**

public 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: " + message);

}

}

**TestLogger.java**

public class TestLogger

{

public static void main(String[] args)

{

Logger logger1 = Logger.*getInstance*();

logger1.log("First log message");

Logger logger2 = Logger.*getInstance*();

logger2.log("Second log message");

if (logger1 == logger2)

{

System.*out*.println("Same Logger instance is used!");

}

else

{

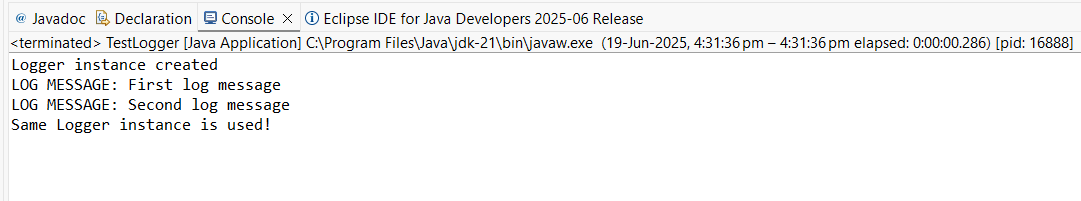
System.*out*.println("Different Logger instances created!");

}

}

}

**Output:**



1. **Implementing the Factory Method Pattern**

**Scenario:**

You are 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.

**Code:**

**Document.java**

public interface Document

{

void openDocument();

}

**WordDocument.java**

public class WordDocument implements Document

{

public void openDocument()

{

System.*out*.println("Opened the Word Document");

}

}

**PdfDocument.java**

public class PdfDocument implements Document

{

public void openDocument()

{

System.*out*.println("Opened the PDF Document");

}

}

**ExcelDocument.java**

public class ExcelDocument implements Document{

public void openDocument()

{

System.*out*.println("Opened the Excel Document");

}

}

**DocumentFactory.java**

public abstract class DocumentFactory {

public abstract Document createDocument();

}

**WordConcreteFactory.java**

public class WordConcreteFactory extends DocumentFactory{

public Document createDocument()

{

return new WordDocument();

}

}

**PdfConcreteFactory.java**

public class PdfConcreteFactory extends DocumentFactory

{

public Document createDocument()

{

return new PdfDocument();

}

}

**ExcelConcreteFactory.java**

public class ExcelConcreteFactory extends DocumentFactory

{

public Document createDocument()

{

return new ExcelDocument();

}

}

**FactoryMethodTest.java**

public class FactoryMethodTest

{

public static void main(String[] args)

{

DocumentFactory word\_doc = new WordConcreteFactory();

Document word = word\_doc.createDocument();

word.openDocument();

DocumentFactory pdf\_doc = new PdfConcreteFactory();

Document pdf = pdf\_doc.createDocument();

pdf.openDocument();

DocumentFactory excel\_doc = new ExcelConcreteFactory();

Document excel = excel\_doc.createDocument();

excel.openDocument();

}

}

**Output:**

