**WEEK 1 HANDS ON EXERCISE - DESIGN PRINCIPLES AND PATTERNS**

**Exercise 1: Singleton Pattern**

**PROBLEM STATEMENT:**

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**

package SingletonPatternExample;

public enum *Logger* {

***Instance***;

// Enum constant, automatically creates only one instance

public void logger(String message)

{

System.***out***.println(message);

}

}

**TestSingleton.java**

package SingletonPatternExample;

public class TestSingleton

{

public static void main(String args[])

{

*Logger* l1 = *Logger*.***Instance***;

*Logger* l2 = *Logger*.***Instance***;

l1.logger("User Logging l1...");

l2.logger("User Logging l2...");

if(l1 == l2)

{

System.***out***.println("Both Logger instances are the same.");

}

else

{

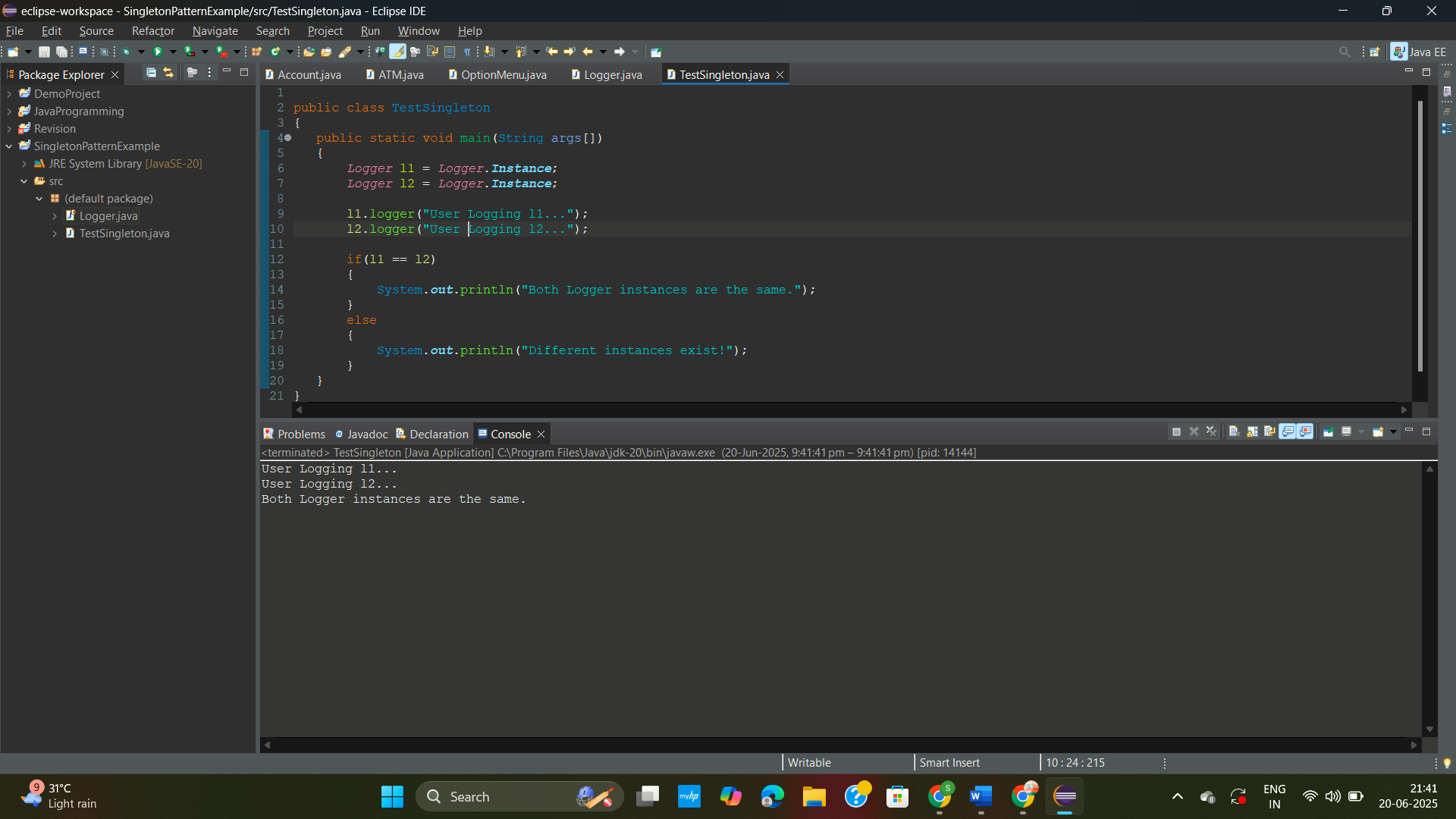
System.***out***.println("Different instances exist!");

}

}

}

**OUTPUT:**



**Exercise 2: Factory Method Pattern**

**PROBLEM STATEMENT:**

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**

package FactoryMethodPatternExample;

public interface Document

{

void open();

}

**ExcelDocument.java**

package FactoryMethodPatternExample;

public class ExcelDocument implements Document

{

*@Override*

public void open()

{

System.***out***.println("\nOpening Excel using MS Excel!..\n");

}

}

**PdfDocument.java**

package FactoryMethodPatternExample;

public class PdfDocument implements Document{

*@Override*

public void open()

{

System.***out***.println("Opening PDF using Adobe Reader!..\n");

}

}

**WordDocument.java**

package FactoryMethodPatternExample;

public class WordDocument implements Document{

*@Override*

public void open()

{

System.***out***.println("Opening Word using MS Word!..\n");

}

}

**DocumentCreator.java**

package FactoryMethodPatternExample;

public abstract class DocumentCreator

{

public abstract Document createDocument();

// we ensure that it must be openable;

}

**ExcelDocumentCreator.java**

package FactoryMethodPatternExample;

public class ExcelDocumentCreator extends DocumentCreator

{

public Document createDocument()

{

return new ExcelDocument();

}

}

**PdfDocumentCreator.java**

package FactoryMethodPatternExample;

public class WordDocumentCreator extends DocumentCreator

{

public Document createDocument()

{

return new WordDocument();

}

}

**WordDocumentCreator.java**

package FactoryMethodPatternExample;

public class WordDocumentCreator extends DocumentCreator

{

public Document createDocument()

{

return new WordDocument();

}

}

**Main.java**

package FactoryMethodPatternExample;

public class Main

{

public static void main(String args[])

{

DocumentCreator creator1 = new ExcelDocumentCreator();

Document excel = creator1.createDocument();

excel.open();

DocumentCreator creator2 = new WordDocumentCreator();

Document word = creator2.createDocument();

word.open();

DocumentCreator creator3 = new PdfDocumentCreator();

Document pdf = creator3.createDocument();

pdf.open();

}

}

**OUTPUT:**

