**Week 1 - Design Patters Principles**

1. **Exercise 1: Implementing the Singleton Pattern**
2. **Define and Implement the Singleton Class**

class Logger {

// Private static instance of Logger (only one instance)

private static Logger instance;

// Private constructor prevents instantiation from other classes

private Logger() {

System.out.println("Logger initialized.");

}

// Public static method to get the instance of Logger

public static Logger getInstance() {

if (instance == null) {

instance = new Logger();

}

return instance;

}

// Sample logging method

public void log(String message) {

System.out.println("Log: " + message);

}

}

1. **Test the Singleton Implementation**

class Main {

public static void main(String[] args) {

// Try to get Logger instance multiple times

Logger logger1 = Logger.getInstance();

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

Logger logger2 = Logger.getInstance();

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

// Check if both references point to the same object

if (logger1 == logger2) {

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

} else {

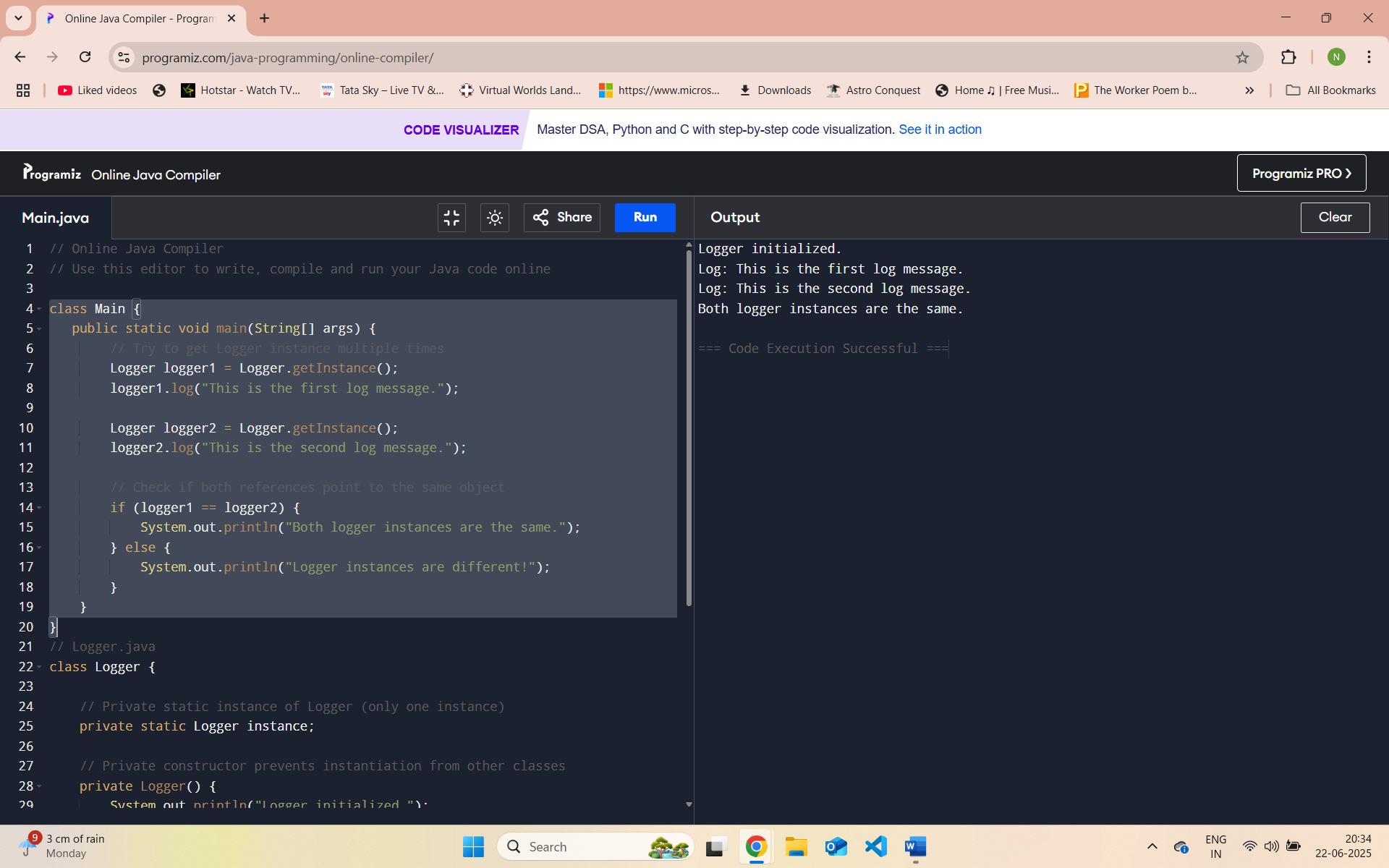
System.out.println("Logger instances are different!");

}

}

}

1. **Output:**



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

**step 1: Create Java Project**

Project Name: FactoryMethodPatternExample

**Step 2: Define Document Interface**

Create an interface named Document.java:

// Document.java

public interface Document {

void open();

}

**Step 3 : Create Concrete Document Classes**

WordDocument.java**:**

// WordDocument.java

public class WordDocument implements Document {

@Override

public void open() {

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

}

}

**PdfDocument.java:**

// PdfDocument.java

public class PdfDocument implements Document {

@Override

public void open() {

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

}

}

**ExcelDocument.java:**

// ExcelDocument.java

public class ExcelDocument implements Document {

@Override

public void open() {

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

}

}

**Step 4: Implement the Factory Method**

DocumentFactory.java (Abstract Factory):

// DocumentFactory.java

public abstract class DocumentFactory {

public abstract Document createDocument();

}

**WordDocumentFactory.java:**

// WordDocumentFactory.java

public class WordDocumentFactory extends DocumentFactory {

@Override

public Document createDocument() {

return new WordDocument();

}

}

**PdfDocumentFactory.java:**

// PdfDocumentFactory.java

public class PdfDocumentFactory extends DocumentFactory {

@Override

public Document createDocument() {

return new PdfDocument();

}

}

**ExcelDocumentFactory.java:**

// ExcelDocumentFactory.java

public class ExcelDocumentFactory extends DocumentFactory {

@Override

public Document createDocument() {

return new ExcelDocument();

}

}

**Step 5: Test the Factory Method Implementation**

Main.java:

// Main.java

public class Main {

public static void main(String[] args) {

// Create a Word Document

DocumentFactory wordFactory = new WordDocumentFactory();

Document wordDoc = wordFactory.createDocument();

wordDoc.open();

// Create a PDF Document

DocumentFactory pdfFactory = new PdfDocumentFactory();

Document pdfDoc = pdfFactory.createDocument();

pdfDoc.open();

// Create an Excel Document

DocumentFactory excelFactory = new ExcelDocumentFactory();

Document excelDoc = excelFactory.createDocument();

excelDoc.open();

}

}

**Output:**

