**Exercise 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.**

**ANS:**

**Logger file:**

public class Logger {

private static Logger instance;

// Private constructor to prevent creating objects from outside

private Logger() {

System.out.println(" The Logger has been created!");

}

// Public method to return the single instance of Logger

public static Logger getInstance() {

if (instance == null) {

instance = new Logger();

}

return instance;

}

public void log(String message) {

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

}

}

**Main file:**

public class Main {

public static void main(String[] args) {

Logger logger1 = Logger.getInstance();

logger1.log("The applicaton is being started now");

// Try getting the Logger again

Logger logger2 = Logger.getInstance();

logger2.log("Still logging !!");

if (logger1 == logger2) {

System.out.println("Yes, both of them are the same Logger instance!");

} else {

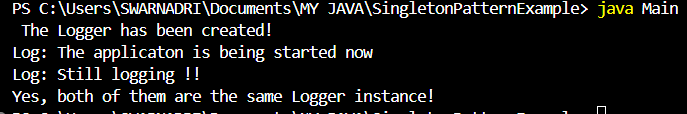
System.out.println("nooooooo");

}

}

}

**OUTPUT:**



**Exercise 2: 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.**

**ANS:**

**word**

public abstract class WordDocument {

public abstract void open();

}

**pdf**

public abstract class PdfDocument {

public abstract void open();

}

**Excel**

public abstract class ExcelDocument {

public abstract void open();

}

**Implemented word**

public class MyWordDocument extends WordDocument {

@Override

public void open() {

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

}

}

**Implemented pdf**

public class MyPdfDocument extends PdfDocument {

@Override

public void open() {

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

}

}

**Implemented excel**

public class MyExcelDocument extends ExcelDocument {

@Override

public void open() {

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

}

}

**Document Factory**

public abstract class DocumentFactory {

public abstract Object createDocument();

}

**Word Factory**

public class WordDocumentFactory extends DocumentFactory {

@Override

public WordDocument createDocument() {

return new MyWordDocument();

}

}

**Pdf Factory**

public class PdfDocumentFactory extends DocumentFactory {

@Override

public PdfDocument createDocument() {

return new MyPdfDocument();

}

}

**Excel Factory**

public class PdfDocumentFactory extends DocumentFactory {

@Override

public PdfDocument createDocument() {

return new MyPdfDocument();

}

}

**Main.java**

public class Main {

public static void main(String[] args) {

WordDocumentFactory wordFactory = new WordDocumentFactory();

WordDocument wordDoc = wordFactory.createDocument();

wordDoc.open();

PdfDocumentFactory pdfFactory = new PdfDocumentFactory();

PdfDocument pdfDoc = pdfFactory.createDocument();

pdfDoc.open();

ExcelDocumentFactory excelFactory = new ExcelDocumentFactory();

ExcelDocument excelDoc = excelFactory.createDocument();

excelDoc.open();

}

}

**OUTPUT:**

