WEEK 1: Design principles & Patterns

**Exercise 1: Implementing the Singleton Pattern**

**Logger.java**

package sample;

public class Logger {

private static Logger instance =null;

private Logger() {

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

}

public static Logger getInstance() {

if (instance == null) {

instance = new Logger();

}

return instance;

}

public void log(String message) {

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

}

}

**Main.java**

package sample;

public class Main

{

public static void main(String[] args) {

Logger log1 = Logger.getInstance();

Logger log2 = Logger.getInstance();

log1.log("User 1");

log2.log("User 2");

If (log1 == log2) {

System.out.println("Only one instance of Logger is created (Singleton)");

} else {

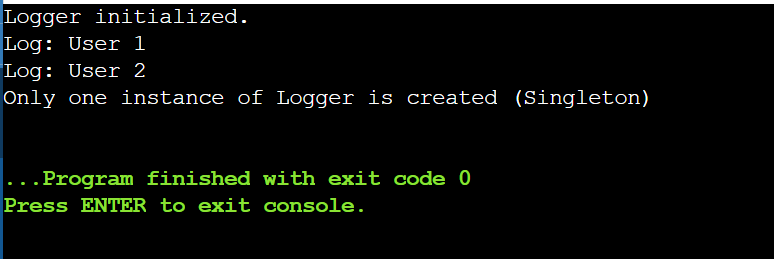
System.out.println("Logger instances are different");

}

}

}

**Output:**



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

**Output:**

**Main.java**

public class Main {

public static void main(String[] args) {

DocumentFactory factory;

factory = new WordDocumentFactory();

Document doc = factory.createDocument();

doc.open();

factory = new PdfDocumentFactory();

doc = factory.createDocument();

doc.open();

factory = new ExcelDocumentFactory();

doc = factory.createDocument();

doc.open();

}

}

**Document.java**

public interface Document {

void open();

}

**WordDocument.java**

public class WordDocument implements Document {

@Override

public void open() {

System.out.println("Opening a Word doc");

}

}

**PdfDocument.java**

public class PdfDocument implements Document {

@Override

public void open() {

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

}

}

**ExcelDocument.java**

public class ExcelDocument implements Document {

@Override

public void open() {

System.out.println("Opening an Excel sheet");

}

}

**DocumentFactory.java**

public abstract class DocumentFactory {

public abstract Document createDocument();

}

**WordDocumentFactory.java**

public class WordDocumentFactory extends DocumentFactory {

@Override

public Document createDocument() {

return new WordDocument();

}

}

**PdfDocumentFactory.java**

public class PdfDocumentFactory extends DocumentFactory {

@Override

public Document createDocument() {

return new PdfDocument();

}

}

**ExcelDocumentFactory.java**

public class ExcelDocumentFactory extends DocumentFactory {

@Override

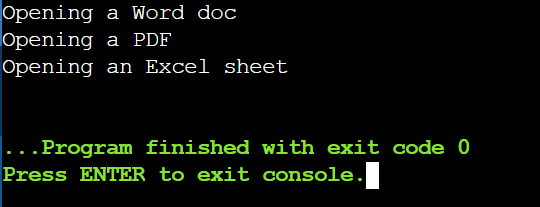
public Document createDocument() {

return new ExcelDocument();

}

}

**Output:**

****