**WEEK – 1 : DESIGN PRINCIPLES AND PATTERN**

**EXERCISE 1 : IMPLEMENTING THE SINGLETON PATTERN**

**Logger.java:**

package com.example.singleton;

public class Logger {

// Private static instance of Logger

private static Logger instance;

// Private constructor to prevent instantiation

private Logger() {

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

}

// Public static method to return the single instance

public static Logger getInstance() {

if (instance == null) {

instance = new Logger();

}

return instance;

}

public void log(String message) {

System.out.println("Person 1 " + message);

}

}

**Singletontest.java:**

package com.example.singleton;

public class SingletonTest {

public static void main(String[] args) {

Logger logger1 = Logger.getInstance();

Logger logger2 = Logger.getInstance();

logger1.log("First message.");

logger2.log("Second message.");

if (logger1 == logger2) {

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

} else {

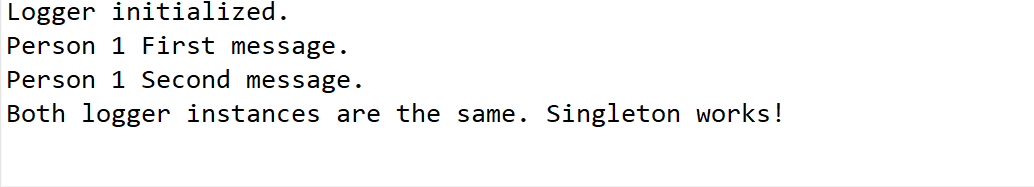
System.out.println("Logger instances are different. Singleton failed.");

}

}

}

**OUTPUT:**



**EXERCISE 2: IMPLEMENTING THE FACTORY METHOD PATTERN**

**Document.java:**

package com.example.factory;

public interface Document {

void open();

}

**WordDocument.java:**

package com.example.factory;

public class WordDocument implements Document {

@Override

public void open() {

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

}

}

**WordDocumentFactory.java:**

package com.example.factory;

public class WordDocumentFactory extends DocumentFactory {

@Override

public Document createDocument() {

return new WordDocument();

}

}

**PdfDocument.java:**

package com.example.factory;

public class PdfDocument implements Document {

@Override

public void open() {

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

}

}

**PdfDocumentFactory.java:**

package com.example.factory;

public class PdfDocumentFactory extends DocumentFactory {

@Override

public Document createDocument() {

return new PdfDocument();

}

}

**ExcelDocument.java:**

package com.example.factory;

public class ExcelDocument implements Document {

@Override

public void open() {

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

}

}

**ExcelDocumentFactory.java:**

package com.example.factory;

public class ExcelDocumentFactory extends DocumentFactory {

@Override

public Document createDocument() {

return new ExcelDocument();

}

}

**DocumentFactory.java:**

package com.example.factory;

public abstract class DocumentFactory {

public abstract Document createDocument();

}

**DocumentFactoryTest.java:**

package com.example.factory;

import java.util.\*;

public class DocumentFactoryTest {

public static void main(String[] args) {

Scanner sc=new Scanner(System.in);

System.out.println("Which document would you like to open? (word/pdf/excel): ");

String input = sc.nextLine().toLowerCase();

switch(input) {

case "word":

DocumentFactory wordFactory = new WordDocumentFactory();

Document wordDoc = wordFactory.createDocument();

wordDoc.open();

break;

case "pdf":

DocumentFactory pdfFactory = new PdfDocumentFactory();

Document pdfDoc = pdfFactory.createDocument();

pdfDoc.open();

break;

case "excel":

DocumentFactory excelFactory = new ExcelDocumentFactory();

Document excelDoc = excelFactory.createDocument();

excelDoc.open();

break;

default:

System.out.println("Invalid input. Please enter 'word', 'pdf', or 'excel'.");

sc.close();

return;

}

sc.close();

}

}

}

**OUTPUT:**

