**Exercise 1: Implementing the Singleton Pattern**

**Logger.java:**

package SingletonPatternExample;

public class Logger{

private static Logger *logger*;

private Logger(){

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

}

public static Logger getInstance(){

if(*logger*==null){

*logger*=new Logger();

}

return *logger*;

}

public void log(String str){

System.*out*.println("Log: "+str);

}

}

**Main.java:**

package SingletonPatternExample;

public class Main {

public static void main(String[] args){

Logger logger1=Logger.*getInstance*();

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

Logger logger2=Logger.*getInstance*();

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

if(logger1==logger2){

System.*out*.println("Both logger1 and logger2 refer to the same instance.");

}

else{

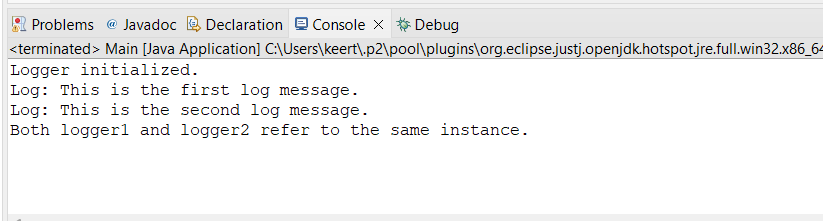
System.*out*.println("Different instances exist!");

}

}

}

**Output:**



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

**WordDocument.java:**

package FactoryMethodPatternExample;

public interface WordDocument {

void open();

}

**PdfDocument.java:**

package FactoryMethodPatternExample;

public interface PdfDocument {

void open();

}

**ExcelDocument.java:**

package FactoryMethodPatternExample;

public interface ExcelDocument {

void open();

}

**WordDocumentImpl.java:**

package FactoryMethodPatternExample;

public class WordDocumentImpl implements WordDocument {

@Override

public void open(){

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

}}

**PdfDocumentImpl.java:**

package FactoryMethodPatternExample;

public class PdfDocumentImpl implements PdfDocument {

@Override

public void open(){

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

}

}

**ExcelDocumentImpl.java:**

package FactoryMethodPatternExample;

public class ExcelDocumentImpl implements ExcelDocument {

@Override

public void open(){

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

}

}

**DocumentFactory.java:**

package FactoryMethodPatternExample;

public abstract class DocumentFactory<T> {

public abstract T createDocument();

}

**WordDocumentFactory.java:**

package FactoryMethodPatternExample;

public class WordDocumentFactory extends DocumentFactory<WordDocument> {

@Override

public WordDocument createDocument(){

return new WordDocumentImpl();

}

}

**PdfDocumentFactory.java:**

package FactoryMethodPatternExample;

public class PdfDocumentFactory extends DocumentFactory<PdfDocument> {

@Override

public PdfDocument createDocument(){

return new PdfDocumentImpl();

}

}

**ExcelDocumentFactory.java:**

package FactoryMethodPatternExample;

public class ExcelDocumentFactory extends DocumentFactory<ExcelDocument> {

@Override

public ExcelDocument createDocument(){

return new ExcelDocumentImpl();

}

}

**Main.java:**

package FactoryMethodPatternExample;

public class Main {

public static void main(String[] args) {

DocumentFactory<WordDocument> wordFactory=new WordDocumentFactory();

WordDocument wordDoc=wordFactory.createDocument();

wordDoc.open();

DocumentFactory<PdfDocument> pdfFactory=new PdfDocumentFactory();

PdfDocument pdfDoc=pdfFactory.createDocument();

pdfDoc.open();

DocumentFactory<ExcelDocument> excelFactory=new ExcelDocumentFactory();

ExcelDocument excelDoc=excelFactory.createDocument();

excelDoc.open();

}

}

**Output:**

