**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.

**Code:**

**Logger.java**

**package** singletonPatternExample;

**public** **class** Logger {

**private** **static** Logger *logger*=**null**;

**private** Logger() {

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

}

**public** **static** Logger getInstance() {

**if**(*logger*==**null**) {

*logger*=**new** Logger();

}

**return** *logger*;

}

**public** **void** display(String str) {

System.***out***.println(str);

}

}

**Test.java**

**package** singletonPatternExample;

**public** **class** Test {

**public** **void** m1() {

Logger.*getInstance*().display("m1 method");

}

**public** **void** m2() {

Logger.*getInstance*().display("m2 method");

}

**public** **static** **void** main(String[] args) {

Logger.*getInstance*().display("main method");

Test t=**new** Test();

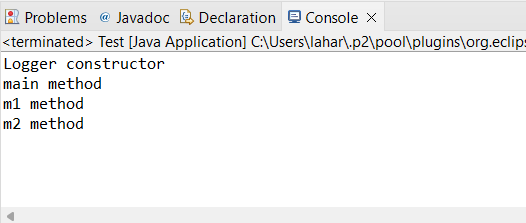
t.m1();

t.m2();

}

}

**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.

**Code:**

**WordDocument.java**

**package** factoryMethodPatternExample;

**public** **interface** WordDocument {

**void** createWord();

}

**PdfDocument.java**

**package** factoryMethodPatternExample;

**public** **interface** PdfDocument {

**void** createPdf();

}

**ExcelDocument.java**

**package** factoryMethodPatternExample;

**public** **interface** ExcelDocument {

**void** createExcel();

}

**WordDocumentImpl.java**

**package** factoryMethodPatternExample;

**public** **class** WordDocumentImpl **implements** WordDocument {

@Override

**public** **void** createWord() {

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

}

}

**PdfDocumentImpl.java**

**package** factoryMethodPatternExample;

**public** **class** PdfDocumentImpl **implements** PdfDocument {

@Override

**public** **void** createPdf() {

System.***out***.println("Creating a Pdf document");

}

}

**ExcelDocumentImpl.java**

**package** factoryMethodPatternExample;

**public** **class** ExcelDocumentImpl **implements** ExcelDocument {

@Override

**public** **void** createExcel() {

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

}

}

**DocumentFactory.java**

**package** factoryMethodPatternExample;

**public** **abstract** **class** DocumentFactory {

**public** **abstract** Object createDocument();

}

**WordDocumentFactory.java**

**package** factoryMethodPatternExample;

**public** **class** WordDocumentFactory **extends** DocumentFactory{

**public** WordDocument createDocument() {

**return** **new** WordDocumentImpl();

}

}

**PdfDocumentFactory.java**

**package** factoryMethodPatternExample;

**public** **class** PdfDocumentFactory **extends** DocumentFactory{

**public** PdfDocument createDocument() {

**return** **new** PdfDocumentImpl();

}

}

**ExcelDocumentFactory.java**

**package** factoryMethodPatternExample;

**public** **class** ExcelDocumentFactory **extends** DocumentFactory{

**public** ExcelDocument createDocument() {

**return** **new** ExcelDocumentImpl();

}

}

**Test.java**

**package** factoryMethodPatternExample;

**public** **class** Test {

**public** **static** **void** main(String[] args) {

WordDocumentFactory wordFactory=**new** WordDocumentFactory();

WordDocument word=wordFactory.createDocument();

word.createWord();

PdfDocumentFactory PdfFactory=**new** PdfDocumentFactory();

PdfDocument Pdf=PdfFactory.createDocument();

Pdf.createPdf();

ExcelDocumentFactory ExcelFactory=**new** ExcelDocumentFactory();

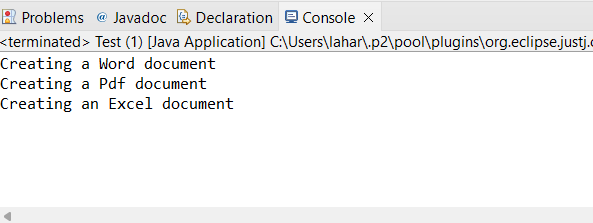
ExcelDocument Excel=ExcelFactory.createDocument();

Excel.createExcel();

}

}

**Output:**

****