WEEK 1 DESIGN PATTERN AND PRINCIPLE

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

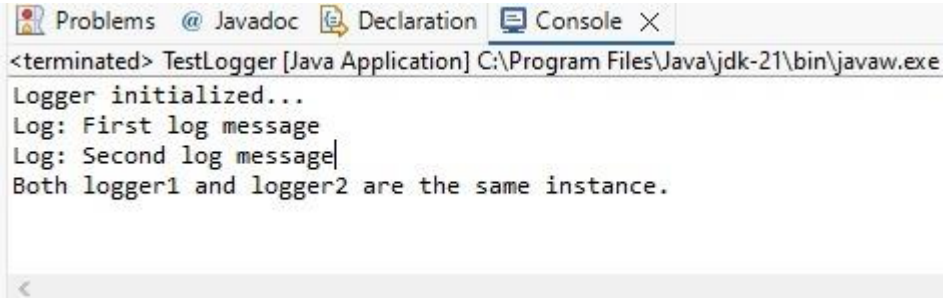
```
package singleton;

public class Logger {

        private static Logger instance;

   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);

   }

}
```

TESTLOGGER.JAVA

```
package singleton;

public class TestLogger {

        public static void main(String[] args) {

            Logger logger1 = Logger.getInstance();

            logger1.log("First log message");

            Logger logger2 = Logger.getInstance();

            logger2.log("Second log message");
```

```
        if (logger1 == logger2) {
            System.out.println("Both logger1 and logger2 are the same instance.");
        } else {
            System.out.println("Different instances exist!");
        }
    }
}
```

OUTPUT



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:

DOCUMENT.JAVA

```
package factory;

public interface Document {
    void open();
}
```

WORDDOCUMENT.JAVA

```
package factory;


public class WordDocument implements Document{
@Override
  public void open() {
     System.out.println("Opening a Word document.");
  }
```

}

PDFDOCUMENT.JAVA

```
package factory;

public class PdfDocument  implements Document{
@Override

  public void open() {
System.out.println("Opening a PDF document.");

   }

}
```

EXCELDOCUMENT.JAVA

```
package factory;

public class ExcelDocument implements Document{

@Override

    public void open() {

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

    }

}
```

DOCUMENTFACTORY.JAVA

```
package factory;

public abstract class DocumentFactory {

 public abstract Document createDocument();

}
```

WORDDOCUMENTFACTORY.JAVA

```
package factory;

public class WordDocumentFactory extends DocumentFactory{

@Override

    public Document createDocument() {

        return new WordDocument();

    }

}
```

PDFDOCUMENTFACTORY.JAVA

```
package factory;
```

```
public class pdfDocumentFactory extends DocumentFactory{
@Override
 public Document createDocument() {
return new PdfDocument();
}
}
```

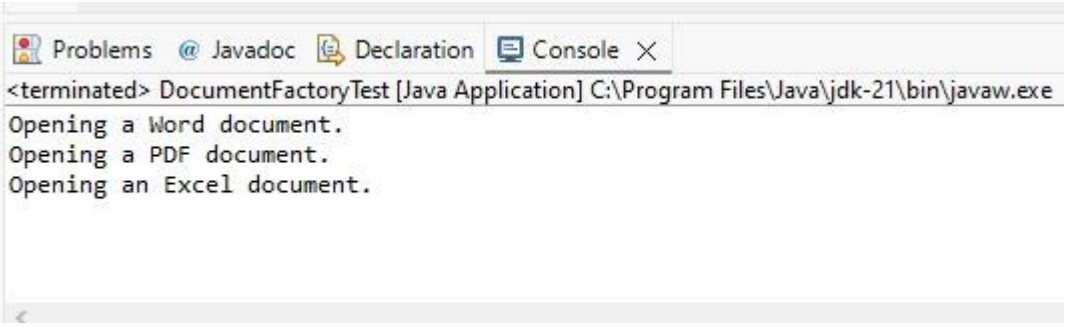
EXCELDOCUMENTFACTORY.JAVA

```
package factory;
public class ExcelDocumentFactory extends DocumentFactory{
@Override
   public Document createDocument() {
      return new ExcelDocument();
   }
}
```

DOCUMENTFACTORYTEST.JAVA

```
package factory;

public class DocumentFactoryTest {
public static void main(String[] args) {
      DocumentFactory wordFactory = new WordDocumentFactory();
      Document wordDoc = wordFactory.createDocument();
      wordDoc.open();
      DocumentFactory pdfFactory = new pdfDocumentFactory();
      Document pdfDoc = pdfFactory.createDocument();
      pdfDoc.open();
      DocumentFactory excelFactory = new ExcelDocumentFactory();
      Document excelDoc = excelFactory.createDocument();
      excelDoc.open();
   }
}
```

OUTPUT


Problems
Javadoc
Declaration
Console
<terminated> DocumentFactoryTest [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe
Opening a Word document.
Opening a PDF document.
Opening an Excel document.