WEEK -01

DESIGN PATTERN AND PRINCIPLES

- **Exercise 1: Implementing the Singleton Pattern**

➔CODE:

Logger.java

```
public class Logger {
  private static Logger instance;

  private Logger() {

  }

  public static Logger getInstance() {
    if (instance == null) {
      instance = new Logger();
    }
    return instance;
  }

  public void log(String msg) {
    System.out.println("log: " + msg);
  }
}
```

Test.java

```
public class Test {
  public static void main(String[] args) {
    Logger lg1 = Logger.getInstance();

    Logger lg2 = Logger.getInstance();

    if(lg1 == lg2){
      System.out.println("\nSame instance\n");
    } else {
      System.out.println("\ndifferent instance\n");
    }

    lg1.log("check your inbox\n");
  }
}
```

O/P:

```
PS C:\Users\schow\Desktop\cts dn 4.0\Deepskilling\solution\week _01> javac SingletonPatternExample/Test.java
PS C:\Users\schow\Desktop\cts dn 4.0\Deepskilling\solution\week _01> java SingletonPatternExample/Test

Same instance

log: check your inbox
```

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

➔CODE:

Document.java

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

WordDocument.java

```
public class WordDocument implements Document {
  @Override
  public void open() {
    System.out.println("Word is opening\n");
  }

  @Override
  public void close() {
    System.out.println("Word is closing\n");
  }
}
```

PdfDocument.java

```
public class PdfDocument implements Document{
  @Override
  public void open() {
    System.out.println("Pdf is opening\n");
  }

  @Override
  public void close() {
    System.out.println("Pdf is closing\n");
  }
}
```

ExcelDocument.java

```
public class ExcelDocument implements Document{
  @Override
  public void open() {
    System.out.println("Excel is opening\n");
  }

  @Override
  public void close() {
    System.out.println("Excel is closing\n");
  }
}
```

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();
  }
}
```

Test.java

```
public class Test {
  public static void main(String[] args) {
    WordDocumentFactory wordFactory = new WordDocumentFactory();
    Document word = wordFactory.createDocument();
    word.open();
    word.close();

    PdfDocumentFactory pdfFactory = new PdfDocumentFactory();
    Document pdf = pdfFactory.createDocument();
    pdf.open();
    pdf.close();

    ExcelDocumentFactory excelFactory = new ExcelDocumentFactory();
    Document excel = excelFactory.createDocument();
    excel.open();
    excel.close();
  }
}
```

O/P:

```
PS C:\Users\schow\Desktop\cts dn 4.0\Deepskilling\solution\week _01> javac FactoryMethodPatternExample/Test.java
PS C:\Users\schow\Desktop\cts dn 4.0\Deepskilling\solution\week _01> java FactoryMethodPatternExample/Test
Word is opening

Word is closing

Pdf is opening

Pdf is closing

Excel is opening

Excel is closing
```