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

**Problem Statement**

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:**

**package Week1\_DesignPatternsAndPrinciples;**

**interface Document {**

**void open();**

**}**

**class WordDocument implements Document {**

**public void open() {**

**System.out.println("Opening Word Document");**

**}**

**}**

**class PdfDocument implements Document {**

**@Override**

**public void open() {**

**System.out.println("Opening PDF Document");**

**}**

**}**

**class ExcelDocument implements Document {**

**public void open() {**

**System.out.println("Opening Excel Document");**

**}**

**}**

**abstract class DocumentFactory {**

**public abstract Document createDocument();**

**}**

**class WordDocumentFactory extends DocumentFactory {**

**@Override**

**public Document createDocument() {**

**return new WordDocument();**

**}**

**}**

**class PdfDocumentFactory extends DocumentFactory {**

**@Override**

**public Document createDocument() {**

**return new PdfDocument();**

**}**

**}**

**class ExcelDocumentFactory extends DocumentFactory {**

**@Override**

**public Document createDocument() {**

**return new ExcelDocument();**

**}**

**}**

**public class FactoryMethodPattern1 {**

**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 Screenshot:**

**A screenshot of a computer

AI-generated content may be incorrect.**