**DESIGN PATTERNS AND PRINCIPLES:**

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

**Step 1: Define the Document Interface:**

**Document.java**

public interface Document {

void open();

}

**Step 2: Create Concrete Document Classes**

**WordDocument.java**

public class WordDocument implements Document {

public void open() {

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

}

}

**// File: PdfDocument.java**

public class PdfDocument implements Document {

public void open() {

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

}

}

**// File: ExcelDocument.java**

public class ExcelDocument implements Document {

public void open() {

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

}

}

**Step 3: Create Abstract Factory:**

**// File: DocumentFactory.java**

public abstract class DocumentFactory {

public abstract Document createDocument();

}

**Step 4: Create Concrete Factories:**

**// File: WordFactory.java**

public class WordFactory extends DocumentFactory {

public Document createDocument() {

return new WordDocument();

}

}

**// File: PdfFactory.java**

public class PdfFactory extends DocumentFactory {

public Document createDocument() {

return new PdfDocument();

}

}

**// File: ExcelFactory.java**

public class ExcelFactory extends DocumentFactory {

public Document createDocument() {

return new ExcelDocument();

}

}

**Step 5: Test the Factory Pattern:**

**// File: TestFactoryPattern.java**

public class TestFactoryPattern {

public static void main(String[] args) {

DocumentFactory wordFactory = new WordFactory();

Document word = wordFactory.createDocument();

word.open(); // Output: Opening Word Document.

DocumentFactory pdfFactory = new PdfFactory();

Document pdf = pdfFactory.createDocument();

pdf.open(); // Output: Opening PDF Document.

DocumentFactory excelFactory = new ExcelFactory();

Document excel = excelFactory.createDocument();

excel.open(); // Output: Opening Excel Document.

}

}

**OUTPUT:**

