**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.

**Steps:**

**1. Create a New Java Project:**

* **Project Name**: FactoryMethodPatternExample
* Use your preferred IDE (like IntelliJ IDEA, Eclipse, or NetBeans) to create a new Java project.

**2. Define Document Classes:**

**Document Interface**:

* Define an interface or an abstract class that represents a generic document.

public interface Document {

void open();

void close();

}

**3. Create Concrete Document Classes:**

**WordDocument Class**:

* A concrete class that implements the Document interface.

public class WordDocument implements Document {

@Override

public void open() {

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

}

@Override

public void close() {

System.out.println("Closing Word document.");

}

}

**PdfDocument Class**:

* A concrete class that implements the Document interface.

public class PdfDocument implements Document {

@Override

public void open() {

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

}

@Override

public void close() {

System.out.println("Closing PDF document.");

}

}

**ExcelDocument Class**:

* A concrete class that implements the Document interface.

public class ExcelDocument implements Document {

@Override

public void open() {

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

}

@Override

public void close() {

System.out.println("Closing Excel document.");

}

}

**4. Implement the Factory Method:**

**DocumentFactory Abstract Class**:

* Define an abstract factory class with a method createDocument().

public abstract class DocumentFactory {

public abstract Document createDocument();

}

**Concrete Factory Classes**:

* Create concrete factory classes for each document type that extend DocumentFactory.

public class WordDocumentFactory extends DocumentFactory {

@Override

public Document createDocument() {

return new WordDocument();

}

}

public class PdfDocumentFactory extends DocumentFactory {

@Override

public Document createDocument() {

return new PdfDocument();

}

}

public class ExcelDocumentFactory extends DocumentFactory {

@Override

public Document createDocument() {

return new ExcelDocument();

}

}

**5. Test the Factory Method Implementation:**

**TestFactoryMethod Class**:

* Create a test class to demonstrate the creation of different document types using the factory method.

public class TestFactoryMethod {

public static void main(String[] args) {

// Create a factory for Word documents

DocumentFactory wordFactory = new WordDocumentFactory();

Document wordDoc = wordFactory.createDocument();

wordDoc.open();

wordDoc.close();

// Create a factory for PDF documents

DocumentFactory pdfFactory = new PdfDocumentFactory();

Document pdfDoc = pdfFactory.createDocument();

pdfDoc.open();

pdfDoc.close();

// Create a factory for Excel documents

DocumentFactory excelFactory = new ExcelDocumentFactory();

Document excelDoc = excelFactory.createDocument();

excelDoc.open();

excelDoc.close();

}

}