Implementing the Factory Method Pattern

Factory Method Pattern:

- The Factory Method Pattern is a creational design pattern that provides an interface for creating objects in a superclass but allows subclasses to alter the type of objects that will be created.
- It helps in encapsulating object creation logic, making the code more maintainable, extensible, and decoupled.
- Instead of calling new directly for each type of document (Word, PDF, Excel), the Factory Method lets the system decide which object to instantiate at runtime — making the code flexible and open to future additions.

Code Components :

1. Document Interface :

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

2. Concrete Document Classes

```
public class WordDocument implements Document { ... }
public class PdfDocument implements Document { ... }
public class ExcelDocument implements Document { ... }
```

Each class implements the Document interface and defines its own version of the open() method.

3. Abstract Factory :

```
public abstract class DocumentFactory {
    public abstract Document createDocument();
}
```

This defines the factory method that must be implemented by all concrete factories.

4. Concrete Factories :

```
public class WordDocumentFactory extends DocumentFactory {
    public Document createDocument() {
    return new WordDocument();
    }
}
```

Each concrete factory knows how to create one specific document type.

5. Client Code – TestDocumentFactory :

```
DocumentFactory factory = new PdfDocumentFactory();
Document doc = factory.createDocument();
doc.open();
```

The client chooses which factory to use at runtime. This allows object creation without knowing the exact class being instantiated.

Benefits Observed in This Implementation :

- Easy to extend — to add a new document type, just create a new class and a factory for it.
- Adheres to the **Open/Closed Principle (** open for extension ans closed for modification**)** — open for extension, closed for modification.
- A proper understanding of abstraction and polymorphism.
- Clean separation of object creation logic.
- Flexibility to scale the application as more document types are added.