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

PROJECT NAME: FactoryMethodPatternExample  
INTERFACE NAME: Document  
Document.java:  
public interface Document {

void open();

}

CLASS NAME: WordDocument  
WordDocument.java:  
public class WordDocument implements Document {

@Override

public void open() {

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

}

}

CLASS NAME: PdfDocument  
PdfDocument.java:  
public class PdfDocument implements Document {

@Override

public void open() {

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

}

}

CLASS NAME: ExcelDocument  
ExcelDocument.java:  
public class ExcelDocument implements Document {

@Override

public void open() {

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

}

}

CLASS NAME: DocumentFactory  
DocumentFactory.java:  
public abstract class DocumentFactory {

public abstract Document createDocument();

}  
  
CLASS NAME: WordDocumentFactory  
WordDocumentFactory.java:  
public class WordDocumentFactory extends DocumentFactory {

@Override

public Document createDocument() {

return new WordDocument();

}

}

CLASS NAME: PdfDocumentFactory  
PdfDocumentFactory.java:  
public class PdfDocumentFactory extends DocumentFactory {

@Override

public Document createDocument() {

return new PdfDocument();

}

}  
  
CLASS NAME: ExcelDocumentFactory  
ExcelDocumentFactory.java:  
public class ExcelDocumentFactory extends DocumentFactory {

@Override

public Document createDocument() {

return new ExcelDocument();

}

}

CLASS NAME: FactoryTest  
FactoryTest.java:  
public class FactoryTest {

public static void main(String[] args) {

DocumentFactory wordFactory = new WordDocumentFactory();

Document word = wordFactory.createDocument();

word.open();

DocumentFactory pdfFactory = new PdfDocumentFactory();

Document pdf = pdfFactory.createDocument();

pdf.open();

DocumentFactory excelFactory = new ExcelDocumentFactory();

Document excel = excelFactory.createDocument();

excel.open();

}

}

**OUTPUT:**

