**DESIGN PATTERNS AND PRINCIPLES**

Exercise 2 : Implementing the Factory Pattern

**Code :**

**Document.java**

public interface Document {  
 void open();  
}  
class WordDocument implements Document{  
 public void open()  
 {  
 System.*out*.println("Opening Word Document....");  
 }  
}  
class PdfDocument implements Document{  
 public void open()  
 {  
 System.*out*.println("Opening PDF Document....");  
 }  
}  
class ExcelDocument implements Document{  
 public void open()  
 {  
 System.*out*.println("Opening Excel Document....");  
 }  
}

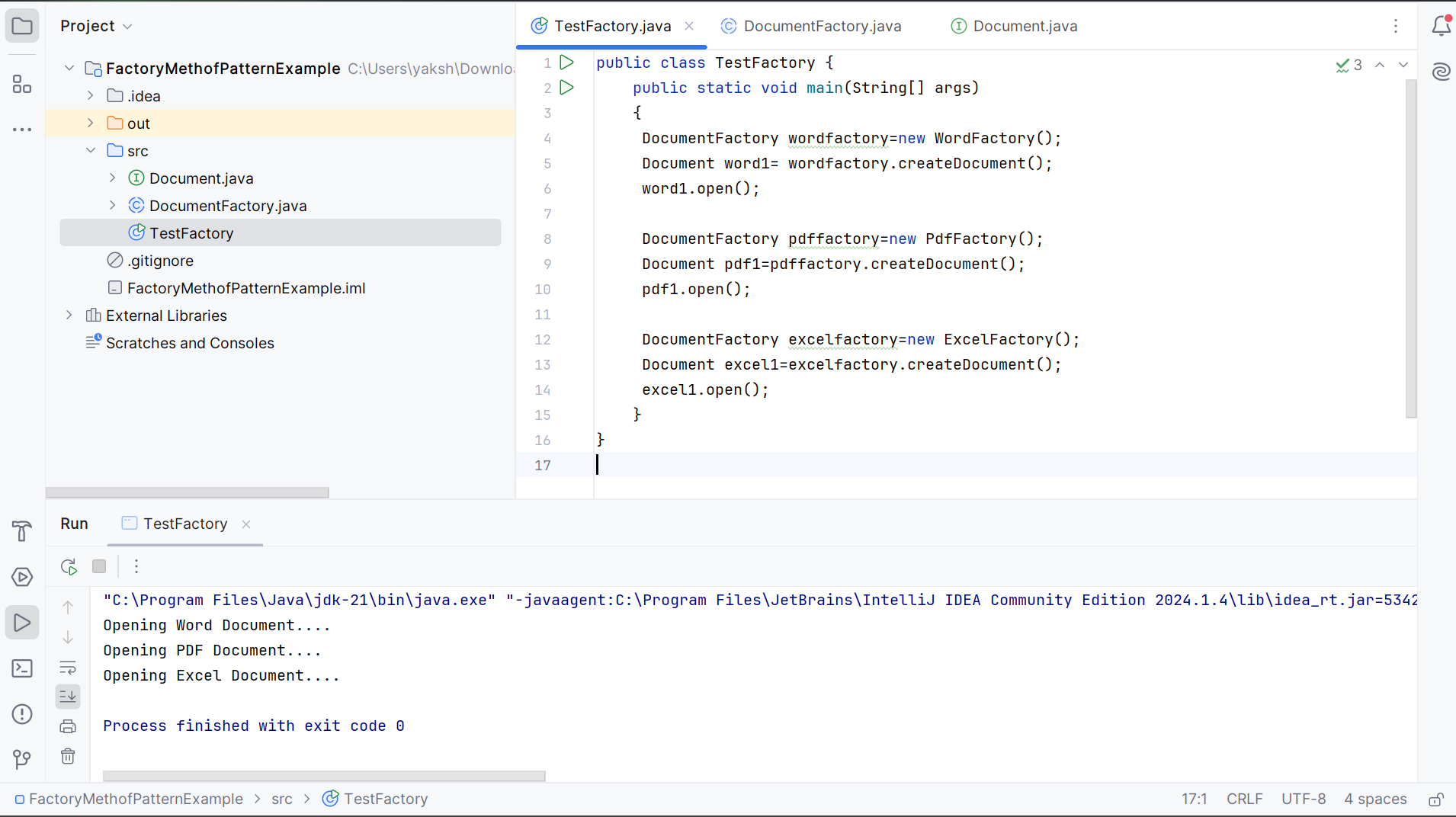
**DocumentFactory.java**

abstract class DocumentFactory {  
 abstract Document createDocument();  
}  
  
class WordFactory extends DocumentFactory{  
 public Document createDocument()  
 {  
 return new WordDocument();  
 }  
}  
class PdfFactory extends DocumentFactory{  
 public Document createDocument()  
 {  
 return new PdfDocument();  
 }  
}  
class ExcelFactory extends DocumentFactory{  
 public Document createDocument()  
 {  
 return new ExcelDocument();  
 }  
}

**TestFactory.java**

public class TestFactory {  
 public static void main(String[] args)  
 {  
 DocumentFactory wordfactory=new WordFactory();  
 Document word1= wordfactory.createDocument();  
 word1.open();  
  
 DocumentFactory pdffactory=new PdfFactory();  
 Document pdf1=pdffactory.createDocument();  
 pdf1.open();  
  
 DocumentFactory excelfactory=new ExcelFactory();  
 Document excel1=excelfactory.createDocument();  
 excel1.open();  
 }  
}

**Output Screenshot:**

****

Thus, different document types have been created using the factory pattern successfully.