**Exercise 1: Implementing the Singleton Pattern**

**Scenario:**

You need to ensure that a logging utility class in your application has only one instance throughout the application lifecycle to ensure consistent logging.

**Project: SingletonPatternExample**

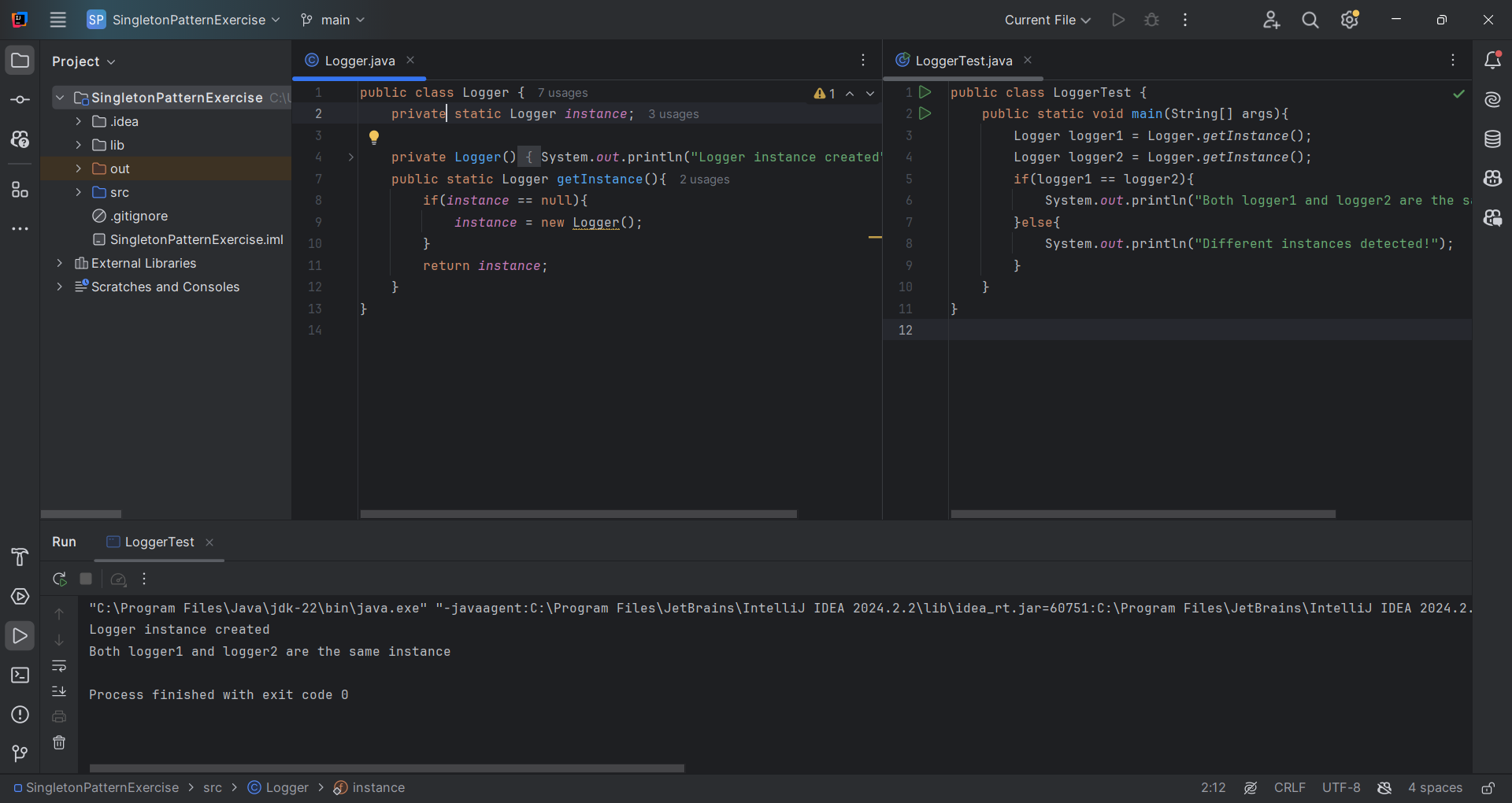
**Logger.java**

public class Logger {  
 private static Logger *instance*;  
  
 private Logger(){  
 System.*out*.println("Logger instance created");  
 }  
 public static Logger getInstance(){  
 if(*instance* == null){  
 synchronized(Logger.class){  
 if(*instance* == null){  
 *instance* = new Logger();  
 }  
 }  
 }  
 return *instance*;  
 }  
}

**LoggerTest.java**

public class LoggerTest {  
 public static void main(String[] args){  
 Logger logger1 = Logger.*getInstance*();  
 Logger logger2 = Logger.*getInstance*();  
   
 if(logger1 == logger2){  
 System.*out*.println("Both logger1 and logger2 are the same instance");  
 }else{  
 System.*out*.println("Different instances detected!");  
 }  
 }  
}

**Output:**



**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.

**Project: FactoryMethodPatternExample:**

**Document.java:**

public interface Document {  
 void message();  
}  
  
class WordDocument implements Document{  
 @Override  
 public void message(){  
 System.*out*.println("Word Document created");  
 }  
}  
  
class PdfDocument implements Document{  
 @Override  
 public void message(){  
 System.*out*.println("Pdf Document created");  
 }  
}  
  
class ExcelDocument implements Document{  
 @Override  
 public void message(){  
 System.*out*.println("Excel Document created");  
 }  
}  
  
abstract class DocumentFactory{  
 public abstract Document createDocument();  
}  
  
class WordDocumentFactory extends DocumentFactory{  
 @Override  
 public Document createDocument(){  
 return new WordDocument();  
 }  
}

class PdfDocumentFactory extends DocumentFactory{  
 @Override  
 public Document createDocument(){  
 return new PdfDocument();  
 }  
}  
  
class ExcelDocumentFactory extends DocumentFactory{  
 @Override  
 public Document createDocument(){  
 return new ExcelDocument();  
 }  
}

**DocumentTest.java**

public class DocumentTest {  
 public static void main(String[] args){  
 DocumentFactory wordFactory = new WordDocumentFactory();  
 Document wordDocument = wordFactory.createDocument();  
 wordDocument.message();  
  
 DocumentFactory pdfFactory = new PdfDocumentFactory();  
 Document pdfDocument = pdfFactory.createDocument();  
 pdfDocument.message();  
  
 DocumentFactory excelFactory = new ExcelDocumentFactory();  
 Document excelDocument = excelFactory.createDocument();  
 excelDocument.message();  
 }  
}

**Output:**

