**Exercise 1: Implementing the Singleton Pattern**

**Code:**

Logger.java

public class Logger {

    private static Logger instance;

    private Logger(){

        System.out.println("Constructor called");

    }

    public static Logger getInstance(){

        if(instance==null){

            instance = new Logger();

        }

        return instance;

    }

    public void log(String message) {

        System.out.println("Log: " + message);

    }

}

App.java

public class App {

    public static void main(String[] args) throws Exception {

        Logger log1= Logger.getInstance();

        Logger log2= Logger.getInstance();

        log1.log("This is a log message from log1.");

        log2.log("This is a log message from log2.");

        if (log1 == log2) {

            System.out.println("Singleton instance.");

        } else {

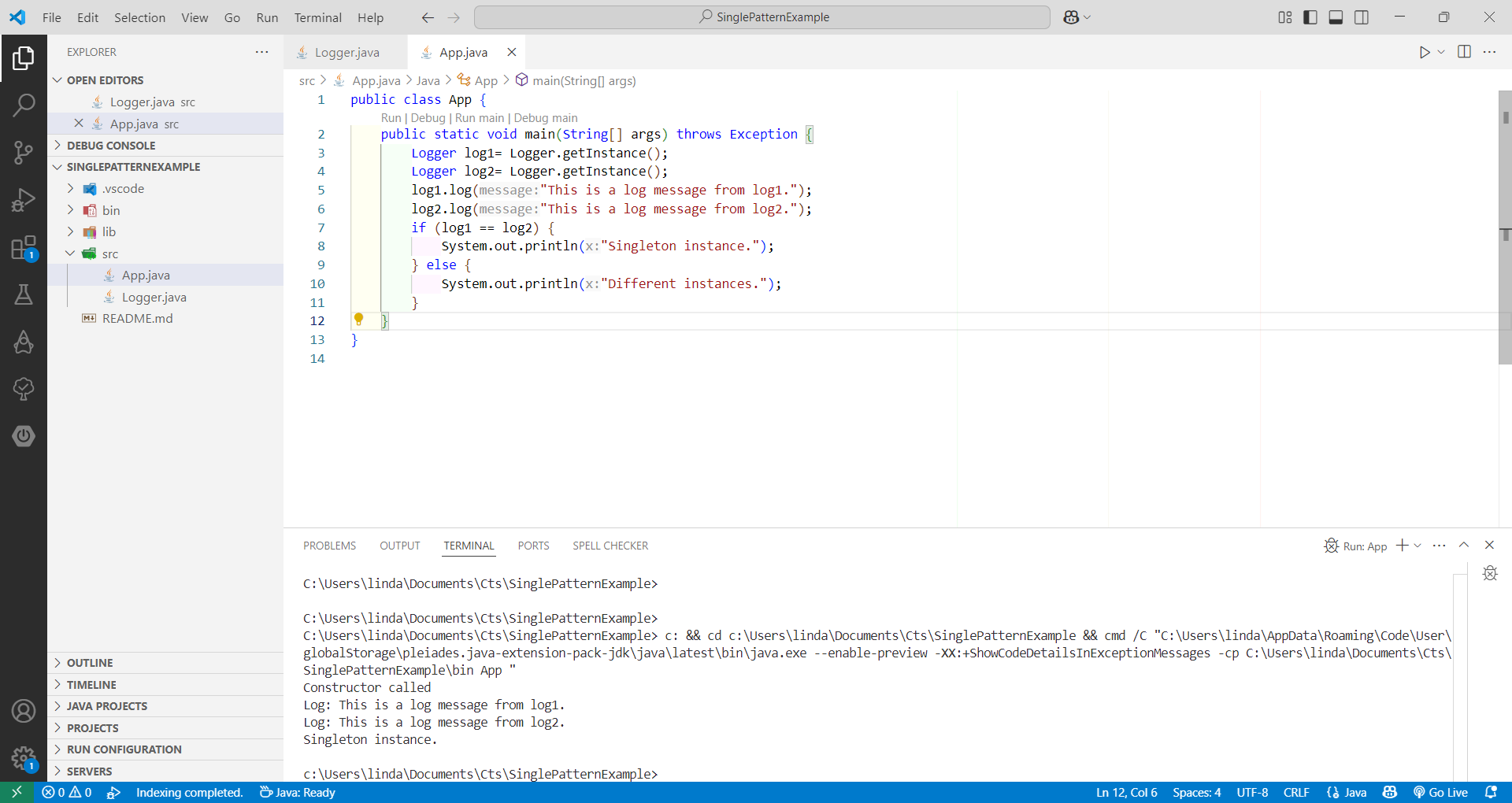
            System.out.println("Different instances.");

        }

    }

}

**Output:**



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

**Code:**

Document.java

public interface Document {

void open();

void close();

void save();

}

WordDocument.java

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");

    }

    @Override

    public void save() {

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

    }

}

PDFDocument.java

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");

    }

    @Override

    public void save() {

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

    }

}

DocumentFactory.java

public abstract class DocumentFactory {

    public abstract Document createDocument();

}

WordDocumentFactory.java

public class WordDocumentFactory extends DocumentFactory {

    @Override

    public Document createDocument() {

        return new WordDocument();

    }

}

PDFDocumentFactory.java

public class PDFDocumentFactory extends DocumentFactory{

    @Override

    public Document createDocument() {

        return new PDFDocument();

    }

}

App.java

public class App {

    public static void main(String[] args) throws Exception {

        System.out.println("");

        DocumentFactory word= new WordDocumentFactory();

        Document wordDoc = word.createDocument();

        wordDoc.open();

        wordDoc.save();

        wordDoc.close();

        System.out.println();

        DocumentFactory pdf = new PDFDocumentFactory();

        Document pdfDoc = pdf.createDocument();

        pdfDoc.open();

        pdfDoc.save();

        pdfDoc.close();

    }

}

**Output:**

