**Mandatory hands-on**

**Exercise 1: Implementing the Singleton Pattern**

**Code:**

// Logger.java

public class Logger {

    private static Logger instance;

    // Private constructor to prevent instantiation

    private Logger() {

        System.out.println("Logger instance created.");

    }

    // Public method to get the instance (lazy initialization)

    public static Logger getInstance() {

        if (instance == null) {

            instance = new Logger();

        }

        return instance;

    }

    // Logging utility method

    public void log(String message) {

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

    }

}

// Main.java

public class Main {

    public static void main(String[] args) {

        Logger logger1 = Logger.getInstance();

        Logger logger2 = Logger.getInstance();

        logger1.log("This is the first log message.");

        logger2.log("This is the second log message.");

        // Check if both references point to the same object

        if (logger1 == logger2) {

            System.out.println("Both logger instances are the same. Singleton verified.");

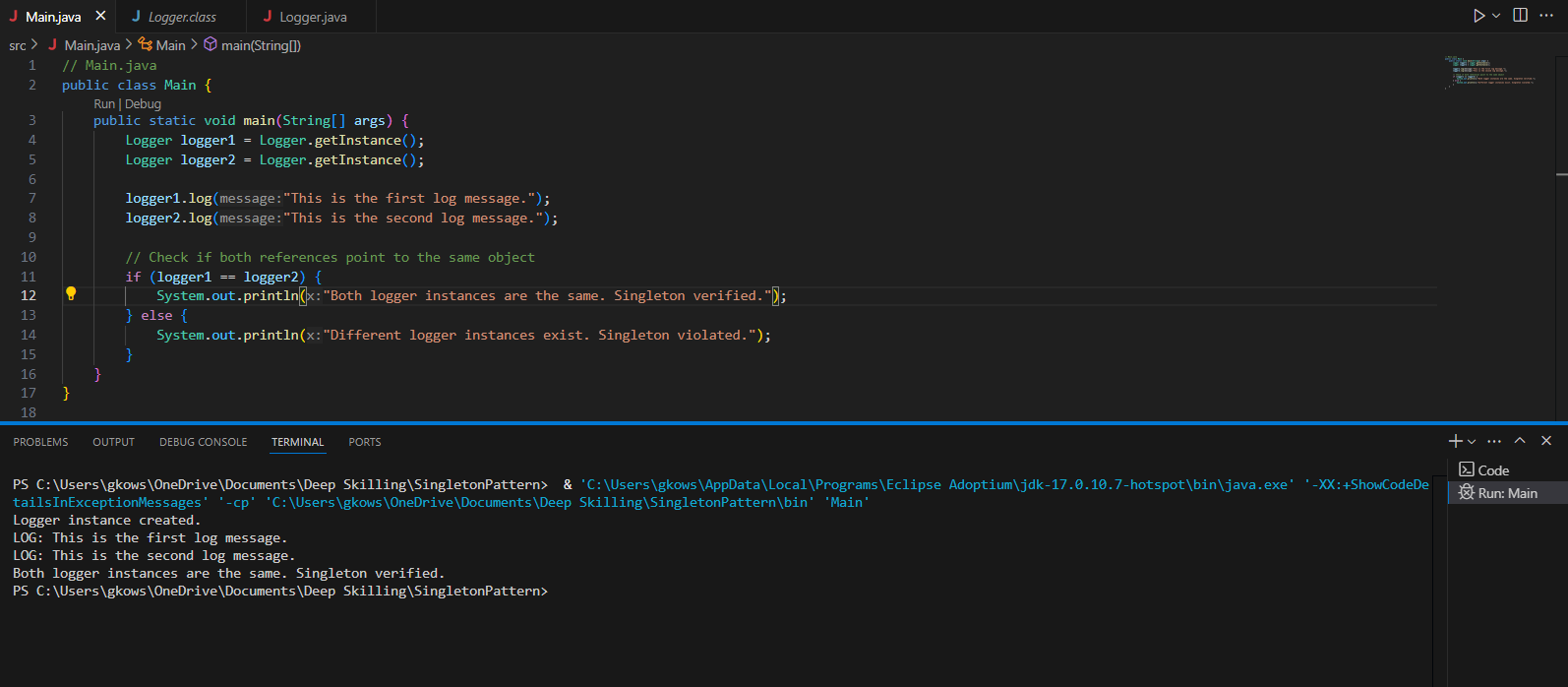
        } else {

            System.out.println("Different logger instances exist. Singleton violated.");

        }

    }

}

**Output:**  


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

**Code:**

// Document.java

public interface Document {

    void open();

}

// PdfDocument.java

public class PdfDocument implements Document {

    @Override

    public void open() {

        System.out.println("Opening a PDF document.");

    }

}

// WordDocument.java

public class WordDocument implements Document {

    @Override

    public void open() {

        System.out.println("Opening a Word document.");

    }

}

// ExcelDocument.java

public class ExcelDocument implements Document {

    @Override

    public void open() {

        System.out.println("Opening an Excel document.");

    }

}

// DocumentFactory.java

public abstract class DocumentFactory {

    public abstract Document createDocument();

}

// PdfDocumentFactory.java

public class PdfDocumentFactory extends DocumentFactory {

    @Override

    public Document createDocument() {

        return new PdfDocument();

    }

}

// WordDocumentFactory.java

public class WordDocumentFactory extends DocumentFactory {

    @Override

    public Document createDocument() {

        return new WordDocument();

    }

}

// ExcelDocumentFactory.java

public class ExcelDocumentFactory extends DocumentFactory {

    @Override

    public Document createDocument() {

        return new ExcelDocument();

    }

}

// Main.java

public class Main {

    public static void main(String[] args) {

// Create and use PDF document

        DocumentFactory pdfFactory = new PdfDocumentFactory();

        Document pdfDoc = pdfFactory.createDocument();

        pdfDoc.open();

        // Create and use Word document

        DocumentFactory wordFactory = new WordDocumentFactory();

        Document wordDoc = wordFactory.createDocument();

        wordDoc.open();

        // Create and use Excel document

        DocumentFactory excelFactory = new ExcelDocumentFactory();

        Document excelDoc = excelFactory.createDocument();

        excelDoc.open();

    }

}

**Output:**

