**Exercise 1: Implementing the Singleton Pattern**

**Code:**

public **class Logger** {

private static Logger instance;

private Logger() {

}

public static Logger getInstance() {

if (instance == null) {

instance = new Logger();

}

return instance;

}

public void log(String message) {

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

}

}

public class **LoggerTest** {

public static void main(String[] args) {

Logger log1 = Logger.getInstance();

Logger log2 = Logger.getInstance();

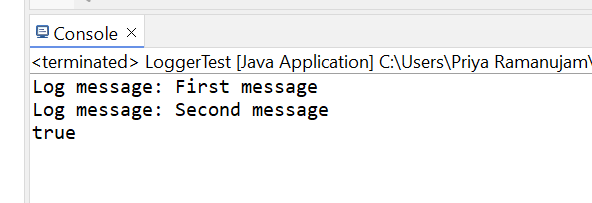
log1.log("First message");

log2.log("Second message");

System.out.println(log1 == log2);

}

**Output:**

****

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

**Code:**

**Document Interface:**

public **interface Document** {

void open();

}

**Concrete Document Classes:**

public **class WordDocument** implements Document {

public void open() {

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

}

}

public **class PdfDocument** implements Document {

public void open() {

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

}

}

public **class ExcelDocument** implements Document {

public void open() {

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

}

}

**Factory Classes:**

public abstract class DocumentFactory {

public abstract Document createDocument();

}

public class WordFactory extends DocumentFactory {

public Document createDocument() {

return new WordDocument();

}

}

public class PdfFactory extends DocumentFactory {

public Document createDocument() {

return new PdfDocument();

}

}

**Test Class:**

public class FactoryTest {

public static void main(String[] args) {

DocumentFactory factory = new WordFactory();

Document doc = factory.createDocument();

doc.open();

}

}

**Output:**

