# Design Pattern and Principles

Exercise1 : Implmenting 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.

## Code:

Logger.java

public class Logger {

private static Logger instance;

private Logger() {

System.out.println("Logger initialized");

}

public static Logger getInstance() { if (instance == null) {

instance = new Logger();

}

return instance;

}

public void log(String message) {

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

}

}

### LoggerTest.java

public class LoggerTest {

public static void main(String[] args) {

Logger logger1 = Logger.getInstance(); logger1.log("First");

Logger logger2 = Logger.getInstance(); logger2.log("Second");

if (logger1 == logger2) {

System.out.println("One instance of Logger.");

} else {

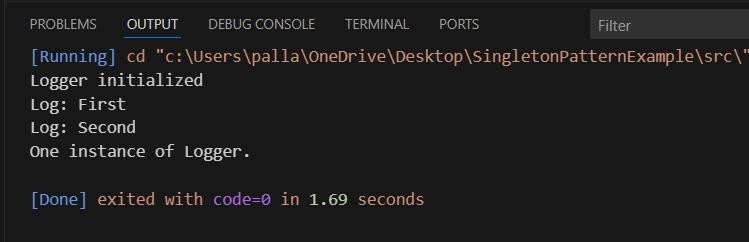
System.out.println("Different instances of Logger.");

}

}

}

# 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.

### Code:

//Document.java

public interface Document { void open();

}

//ExcelDocument.java

public class ExcelDocument implements Document { @Override

public void open() {

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

}

}

//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.");

}

}

//DocumentFactory.java

public abstract class DocumentFactory { public abstract Document createDocument();

}

//ExcelDocumentFactory.java

public class ExcelDocumentFactory extends DocumentFactory { @Override

public Document createDocument() { return new ExcelDocument();

}

}

//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();

}

}

//DcoumentFactoryTest.java

public class DocumentFactoryTest { public static void main(String[] args) {

DocumentFactory wordFactory = new WordDocumentFactory(); Document wordDoc = wordFactory.createDocument();

wordDoc.open();

DocumentFactory pdfFactory = new PdfDocumentFactory(); Document pdfDoc = pdfFactory.createDocument();

pdfDoc.open();

DocumentFactory excelFactory = new ExcelDocumentFactory(); Document excelDoc = excelFactory.createDocument();

excelDoc.open();

}

}

OUTPUT:

