**Superset Id- 6363535**

**Exercise 1: Implementing the Singleton Pattern**

**Code:**

**Logger.cs**

using System;

public class Logger

{

private static Logger instance = null;

private static readonly object lockObj = new object();

private Logger()

{

Console.WriteLine("Logger initialized.");

}

public static Logger GetInstance()

{

if (instance == null)

{

lock (lockObj)

{

if (instance == null)

{

instance = new Logger();

}

}

}

return instance;

}

public void Log(string message)

{

Console.WriteLine($"Log: {message}");

}

}

**Program.cs**

using System;

class Program

{

static void Main(string[] args)

{

Logger logger1 = Logger.GetInstance();

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

Logger logger2 = Logger.GetInstance();

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

if (logger1 == logger2)

{

Console.WriteLine("logger1 and logger2 are the same instance.");

}

else

{

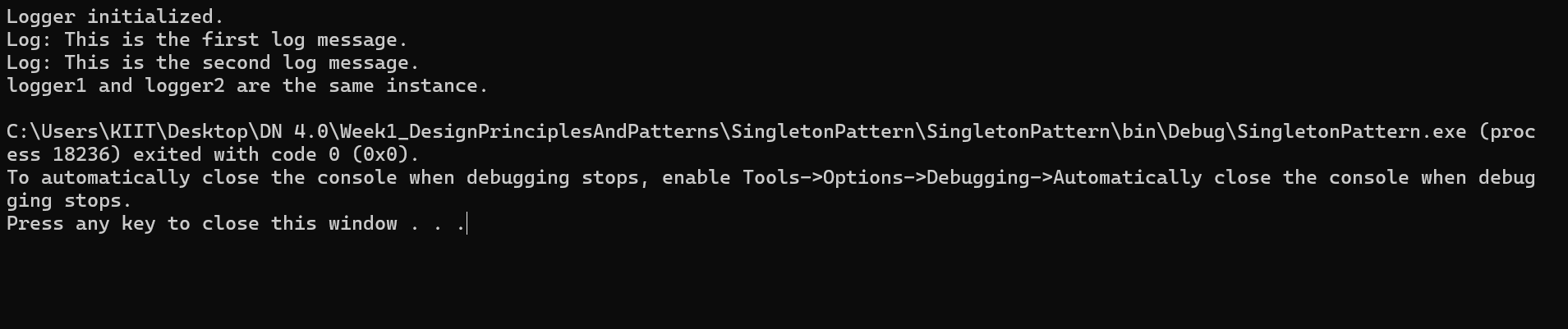
Console.WriteLine("logger1 and logger2 are different instances.");

}

}

}

**Output:**

****

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

**Code:**

[**IDocument.cs**](http://idocument.cs)

public interface IDocument

{

void Open();

}

[**WordDocument.cs**](http://worddocument.cs)

using System;

public class WordDocument : IDocument

{

public void Open()

{

Console.WriteLine("Opening a Word document.");

}

}

[**PdfDocument.cs**](http://pdfdocument.cs)

using System;

public class PdfDocument : IDocument

{

public void Open()

{

Console.WriteLine("Opening a PDF document.");

}

}

[**ExcelDocument.cs**](http://exceldocument.cs)

using System;

public class ExcelDocument : IDocument

{

public void Open()

{

Console.WriteLine("Opening an Excel document.");

}

}

[**DocumentFactory.cs**](http://documentfactory.cs)

public abstract class DocumentFactory

{

public abstract IDocument CreateDocument();

}

[**WordFactory.cs**](http://wordfactory.cs)

public class WordFactory : DocumentFactory

{

public override IDocument CreateDocument()

{

return new WordDocument();

}

}

[**PdfFactory.cs**](http://pdffactory.cs)

public class PdfFactory : DocumentFactory

{

public override IDocument CreateDocument()

{

return new PdfDocument();

}

}

[**ExcelFactory.cs**](http://excelfactory.cs)

public class ExcelFactory : DocumentFactory

{

public override IDocument CreateDocument()

{

return new ExcelDocument();

}

}

[**Program.cs**](http://program.cs)

using System;

class Program

{

static void Main(string[] args)

{

DocumentFactory wordFactory = new WordFactory();

IDocument wordDoc = wordFactory.CreateDocument();

wordDoc.Open();

DocumentFactory pdfFactory = new PdfFactory();

IDocument pdfDoc = pdfFactory.CreateDocument();

pdfDoc.Open();

DocumentFactory excelFactory = new ExcelFactory();

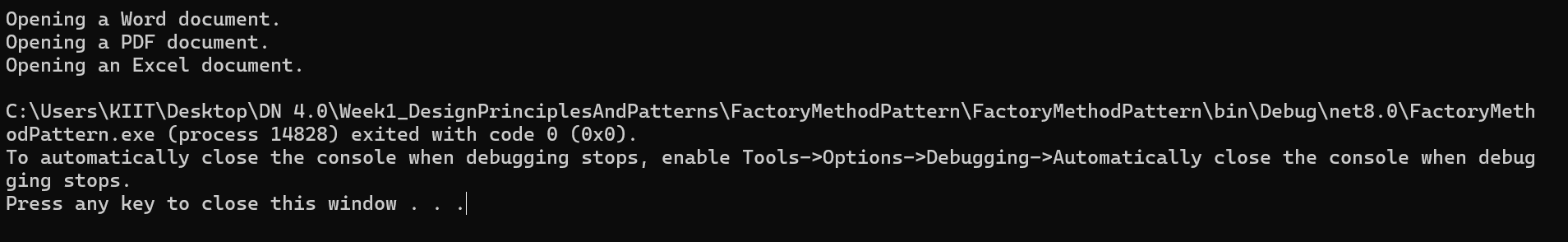
IDocument excelDoc = excelFactory.CreateDocument();

excelDoc.Open();

}

}

**Output:**

****