**Week 1 Design Patterns And Principles**

**Exercise 1: Implementing the Singleton Pattern**

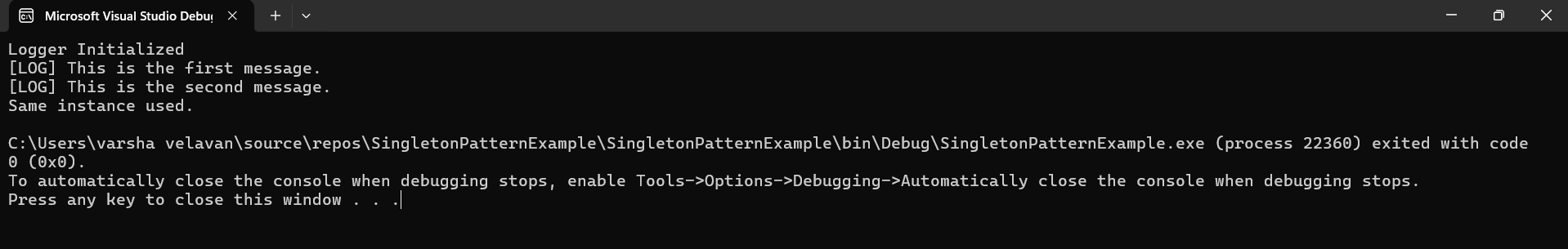
**Code:**

**Logger.cs**

using System;  
  
public class Logger  
{  
 private static Logger \_instance;  
 private static readonly object \_lock = new object();  
  
 private Logger()  
 {  
 Console.WriteLine("Logger Initialized");  
 }  
  
 public static Logger GetInstance()  
 {  
 if (\_instance == null)  
 {  
 lock (\_lock)  
 {  
 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();  
 Logger logger2 = Logger.GetInstance();  
  
 logger1.Log("This is the first message.");  
 logger2.Log("This is the second message.");  
  
 Console.WriteLine(object.ReferenceEquals(logger1, logger2)  
 ? "Same instance used." : "Different instances!");  
 }  
}

**Output:**  


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

**Code:**

**IDocument.cs**

public interface IDocument  
{  
 void Open();  
}

**WordDocument.cs**

using System;  
  
public class WordDocument : IDocument  
{  
 public void Open()  
 {  
 Console.WriteLine("Opening Word Document");  
 }  
}

**PdfDocument.cs**

using System;  
  
public class PdfDocument : IDocument  
{  
 public void Open()  
 {  
 Console.WriteLine("Opening PDF Document");  
 }  
}

**ExcelDocument.cs**

using System;  
  
public class ExcelDocument : IDocument  
{  
 public void Open()  
 {  
 Console.WriteLine("Opening Excel Document");  
 }  
}

**DocumentFactory.cs**

public abstract class DocumentFactory  
{  
 public abstract IDocument CreateDocument();  
}

**WordFactory.cs**

public class WordFactory : DocumentFactory  
{  
 public override IDocument CreateDocument()  
 {  
 return new WordDocument();  
 }  
}

**PdfFactory.cs**

public class PdfFactory : DocumentFactory  
{  
 public override IDocument CreateDocument()  
 {  
 return new PdfDocument();  
 }  
}

**ExcelFactory.cs**

public class ExcelFactory : DocumentFactory  
{  
 public override IDocument CreateDocument()  
 {  
 return new ExcelDocument();  
 }  
}

**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:**  
