**Exercise 1: Implementing the Singleton Pattern**

using System;

public class Logger

{

private static Logger \_instance = null;

private static readonly object \_lock = new object();

// Private constructor prevents external instantiation

private Logger()

{

Console.WriteLine("Logger initialized.");

}

// Public method with lazy initialization and thread-safety

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}");

}

}

public class Program

{

public static void Main(string[] args)

{

Logger logger1 = Logger.GetInstance();

Logger logger2 = Logger.GetInstance();

if (ReferenceEquals(logger1, logger2))

{

Console.WriteLine("Singleton verified: Both instances are the same.");

}

else

{

Console.WriteLine("Singleton failed: Different instances exist.");

}

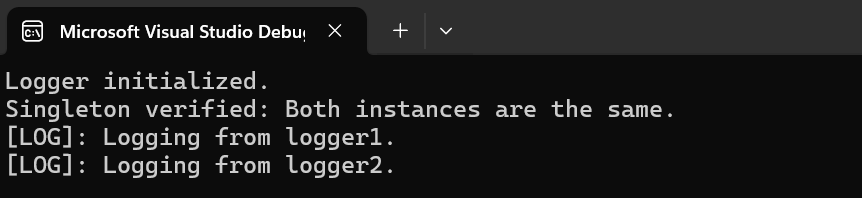
logger1.Log("Logging from logger1.");

logger2.Log("Logging from logger2.");

}

}

**OUTPUT**

****

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

using System;

public interface IDocument

{

void Open();

void Save();

void Close();

}

public class WordDoc : IDocument

{

public void Open() => Console.WriteLine("WordDoc opened.");

public void Save() => Console.WriteLine("WordDoc saved.");

public void Close() => Console.WriteLine("WordDoc closed.");

}

public class PdfDoc : IDocument

{

public void Open() => Console.WriteLine("PdfDoc opened.");

public void Save() => Console.WriteLine("PdfDoc saved.");

public void Close() => Console.WriteLine("PdfDoc closed.");

}

public class ExcelDoc : IDocument

{

public void Open() => Console.WriteLine("ExcelDoc opened.");

public void Save() => Console.WriteLine("ExcelDoc saved.");

public void Close() => Console.WriteLine("ExcelDoc closed.");

}

public abstract class DocumentFactory

{

public abstract IDocument CreateDocument();

}

public class WordDocFactory : DocumentFactory

{

public override IDocument CreateDocument()

{

return new WordDoc();

}

}

public class PdfDocFactory : DocumentFactory

{

public override IDocument CreateDocument()

{

return new PdfDoc();

}

}

public class ExcelDocFactory : DocumentFactory

{

public override IDocument CreateDocument()

{

return new ExcelDoc();

}

}

public class Program

{

public static void Main(string[] args)

{

DocumentFactory[] factories = new DocumentFactory[]

{

new WordDocFactory(),

new PdfDocFactory(),

new ExcelDocFactory()

};

foreach (var factory in factories)

{

IDocument doc = factory.CreateDocument();

doc.Open();

doc.Save();

doc.Close();

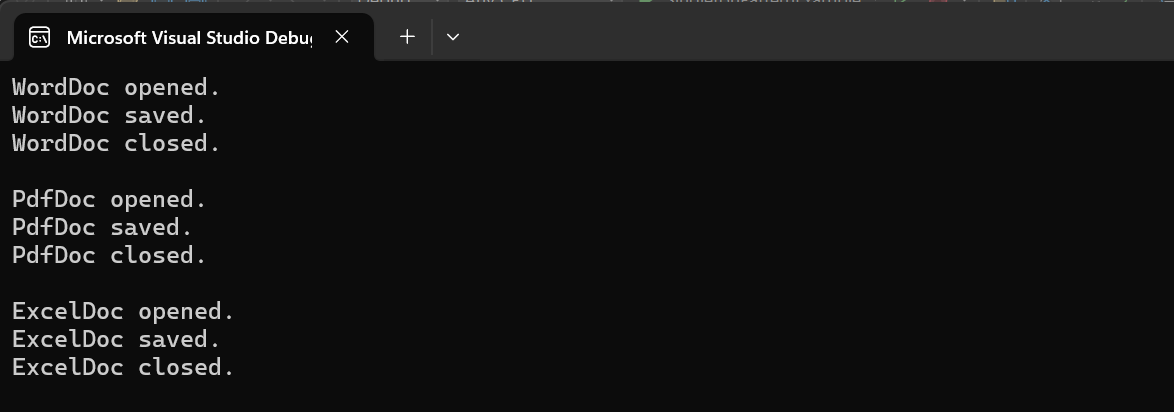
Console.WriteLine();

}

}

}

**OUTPUT**

****