**Design Principles and Patterns**

**Exercise 1: Implementing the Singleton Pattern**

**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")

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

Console.WriteLine("Are both instances same? " + (logger1 == logger2));

    }

}

**Logger.cs:**

using System;

public class Logger

{

private static Logger instance;

private static readonly object lockObj = new object();

private Logger()

{

Console.WriteLine("Logger created");

}

public static Logger GetInstance()

{

if (instance == null)

{

lock (lockObj)

{

if (instance == null)

{

instance = new Logger();

}

}

}

return instance;

}

public void Log(string message)

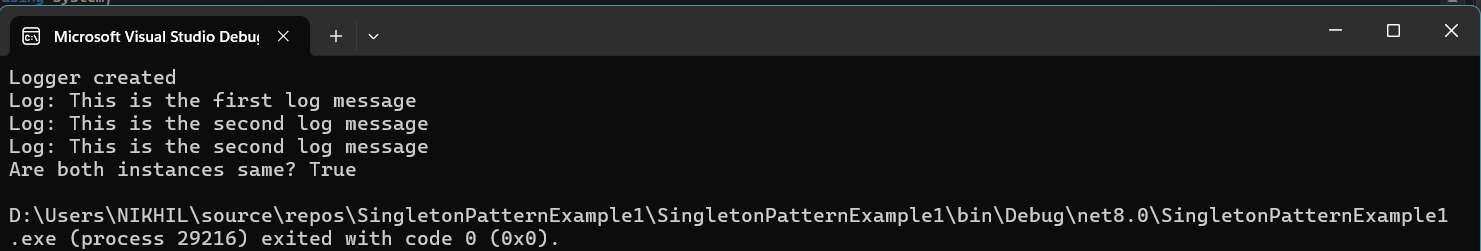
{

Console.WriteLine("Log: " + message);

}

}

**Output:**



**Exercise-2: FactoryMethodPattern**

**Program.cs:**

using System;

class Program

{

static void Main(string[] args)

{

DocumentFactory factory;

factory = new WordFactory();

IDocument word = factory.CreateDocument();

word.Open();

factory = new PdfFactory();

IDocument pdf = factory.CreateDocument();

pdf.Open();

factory = new ExcelFactory();

IDocument excel = factory.CreateDocument();

excel.Open();

}

}

**IDocument.cs:**

public interface IDocument

{

void Open();

}

**Document Factory.cs:**

public abstract class DocumentFactory

{

public abstract IDocument CreateDocument();

}

**Excel Document.cs:**  
public class ExcelDocument : IDocument

{

public void Open()

{

Console.WriteLine("Opening Excel Document...");

}

}

**Excel Factory.cs:**

public class ExcelFactory : DocumentFactory

{

public override IDocument CreateDocument()

{

return new ExcelDocument();

}

}

**Word Document.cs:**

public class WordDocument : IDocument

{

public void Open()

{

Console.WriteLine("Opening Word Document...");

}

}

**Word Factory.cs:**

public class WordFactory : DocumentFactory

{

public override IDocument CreateDocument()

{

return new WordDocument();

}

}

**PDF Document.cs:**

public class PdfDocument : IDocument

{

public void Open()

{

Console.WriteLine("Opening PDF Document...");

}

}

**PDF Factory.cs:**

public class PdfFactory : DocumentFactory

{

public override IDocument CreateDocument()

{

return new PdfDocument();

}

}

**Output:**

