**Exercise 1: Implementing the Singleton Pattern**

using System;

class Logger

{

    private static Logger? \_instance;

cd

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

    // private constructor to prevent external instantiation

    private Logger()

    {

        Console.WriteLine("Logger initialized.");

    }

    // public method to get the single instance

    public static Logger GetInstance()

    {

        if (\_instance == null)

        {

            lock (\_lock) // Thread safety

            {

                if (\_instance == null)

                {

                    \_instance = new Logger();

                }

            }

        }

        return \_instance;

    }

    // method to log messages

    public void Log(string message)

    {

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

    }

}

class Program

{

    static void Main()

    {

        // get the singleton instance

        Logger logger1 = Logger.GetInstance();

        Logger logger2 = Logger.GetInstance();

        // use the logger

        logger1.Log("First message");

        logger2.Log("Second message");

        // check if both instances are the same

        if (ReferenceEquals(logger1, logger2))

        {

            Console.WriteLine("Logger is a singleton. Only one instance exists.");

        }

        else

        {

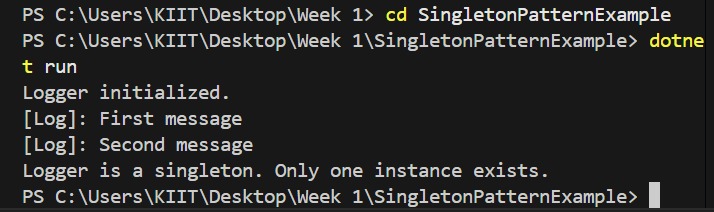
            Console.WriteLine("Logger is NOT a singleton.");

        }

    }

}

**OUTPUT**



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

using System;

// define a common interface for all documents

public interface IDocument

{

    void Open();

}

// create concrete classes for each document type

public class WordDocument : IDocument

{

    public void Open()

    {

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

    }

}

public class PdfDocument : IDocument

{

    public void Open()

    {

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

    }

}

public class ExcelDocument : IDocument

{

    public void Open()

    {

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

    }

}

// define the abstract factory

public abstract class DocumentFactory

{

    public abstract IDocument CreateDocument();

}

// create concrete factories for each document type

public class WordDocumentFactory : DocumentFactory

{

    public override IDocument CreateDocument()

    {

        return new WordDocument();

    }

}

public class PdfDocumentFactory : DocumentFactory

{

    public override IDocument CreateDocument()

    {

        return new PdfDocument();

    }

}

public class ExcelDocumentFactory : DocumentFactory

{

    public override IDocument CreateDocument()

    {

        return new ExcelDocument();

    }

}

// test the factory method implementation

class Program

{

    static void Main()

    {

        DocumentFactory wordFactory = new WordDocumentFactory();

        IDocument word = wordFactory.CreateDocument();

        word.Open();

        DocumentFactory pdfFactory = new PdfDocumentFactory();

        IDocument pdf = pdfFactory.CreateDocument();

        pdf.Open();

        DocumentFactory excelFactory = new ExcelDocumentFactory();

        IDocument excel = excelFactory.CreateDocument();

        excel.Open();

    }

}

**OUTPUT**

