**Q)Implementing the Singleton Pattern**

**Scenario:**

You need to ensure that a logging utility class in your application has only one instance throughout the application lifecycle to ensure consistent logging.

**Steps:**

1. **Create a New Java Project:**
   * Create a new Java project named **SingletonPatternExample**.
2. **Define a Singleton Class:**
   * Create a class named Logger that has a private static instance of itself.
   * Ensure the constructor of Logger is private.
   * Provide a public static method to get the instance of the Logger class.
3. **Implement the Singleton Pattern:**
   * Write code to ensure that the Logger class follows the Singleton design pattern.
4. **Test the Singleton Implementation:**
   * Create a test class to verify that only one instance of Logger is created and used across the application.

using System;

namespace SingletonPatternExample

{

    class Program

    {

        static void Main(string[] args)

        {

            // Test Singleton Behavior

            Logger logger1 = Logger.GetInstance();

            Logger logger2 = Logger.GetInstance();

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

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

            // Verify both references point to the same object

            if (ReferenceEquals(logger1, logger2))

            {

                Console.WriteLine("Both logger1 and logger2 refer to the same instance.");

            }

            else

            {

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

            }

        }

    }

}

using System;

namespace SingletonPatternExample

{

    public class Logger

    {

        private static Logger? \_instance;

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

        private Logger()

        {

            Console.WriteLine("Logger instance created");

        }

        public static Logger GetInstance()

        {

            if (\_instance == null)

            {

                lock (\_lock)

                {

                    if (\_instance == null)

                    {

                        \_instance = new Logger();

                    }

                }

            }

            return \_instance;

        }

        public void Log(string message)

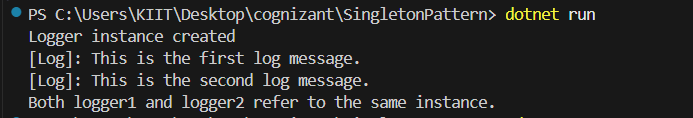
        {

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

        }

    }

}

**Q)Implementing the Factory Method Pattern**

**Scenario:**

You are developing a document management system that needs to create different types of documents (e.g., Word, PDF, Excel). Use the Factory Method Pattern to achieve this.

**Steps:**

1. **Create a New Java Project:**
   * Create a new Java project named **FactoryMethodPatternExample**.
2. **Define Document Classes:**
   * Create interfaces or abstract classes for different document types such as **WordDocument**, **PdfDocument**, and **ExcelDocument**.
3. **Create Concrete Document Classes:**
   * Implement concrete classes for each document type that implements or extends the above interfaces or abstract classes.
4. **Implement the Factory Method:**
   * Create an abstract class **DocumentFactory** with a method **createDocument()**.
   * Create concrete factory classes for each document type that extends DocumentFactory and implements the **createDocument()** method.
5. **Test the Factory Method Implementation:**
   * Create a test class to demonstrate the creation of different document types using the factory method.

using System;

namespace FactoryMethodPatternExample

{

    class Program

    {

        static void Main(string[] args)

        {

            DocumentFactory wordFactory = new WordDocumentFactory();

            IDocument wordDoc = wordFactory.CreateDocument();

            wordDoc.Open();

            DocumentFactory pdfFactory = new PdfDocumentFactory();

            IDocument pdfDoc = pdfFactory.CreateDocument();

            pdfDoc.Open();

            DocumentFactory excelFactory = new ExcelDocumentFactory();

            IDocument excelDoc = excelFactory.CreateDocument();

            excelDoc.Open();

        }

    }

}

