**SUBMITTED BY: SUPERSET ID-6361172**

**Week 1\_Design Patterns and Principles**

**Exercise 1: Implementing the Singleton Pattern**

**CODE:**

**Logger.cs**

//Submitted by:Superset ID: 6361172

//Exercise 1: Implementing the Singleton Pattern

using System;

namespace Singleton\_Pattern

{

public sealed class Logger

{

private static Logger \_instance = null;

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**

//Submitted by:Superset ID: 6361172

//Exercise 1: Implementing the Singleton Pattern

using System;

namespace Singleton\_Pattern

{

class Program

{

static void Main(string[] args)

{

Logger logger1 = Logger.GetInstance();

logger1.Log("First log message");

Logger logger2 = Logger.GetInstance();

logger2.Log("Second log message");

if (ReferenceEquals(logger1, logger2))

{

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

}

else

{

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

}

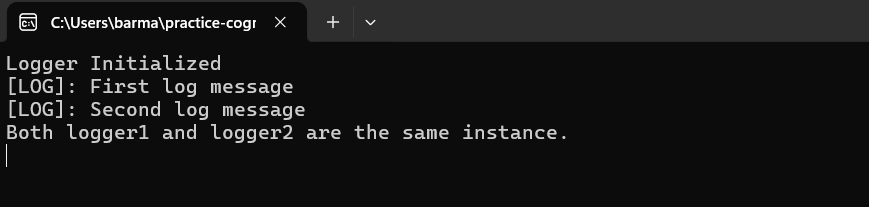
Console.ReadLine();

}

}

}

**OUTPUT:**

****

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

**Program.cs**

//Submitted by: Superset ID- 6361172

//Exercise 2: Implementing the Factory Method Pattern

using System;

namespace FactoryMethodPattern

{

class Program

{

static void Main(string[] args)

{

Console.WriteLine("-----IMPLEMENTATION OF FACTORY METHOD PATTERN-----");

Console.WriteLine("--------------------------------------------------");

DocumentFactory wordFactory = new WordDocumentFactory();

IDocument word = wordFactory.CreateDocument();

word.Open();

word.Close();

Console.WriteLine();

DocumentFactory pdfFactory = new PdfDocumentFactory();

IDocument pdf = pdfFactory.CreateDocument();

pdf.Open();

pdf.Close();

Console.WriteLine();

DocumentFactory excelFactory = new ExcelDocumentFactory();

IDocument excel = excelFactory.CreateDocument();

excel.Open();

excel.Close();

Console.WriteLine("--------------------------");

Console.ReadLine();

}

}

}

**IDocument.cs**

//Submitted by: Superset ID- 6361172

//Exercise 2: Implementing the Factory Method Pattern

namespace FactoryMethodPattern

{

public interface IDocument

{

void Open();

void Close();

}

}

**ExcelDocument.cs**

//Submitted by: Superset ID- 6361172

//Exercise 2: Implementing the Factory Method Pattern

using System;

namespace FactoryMethodPattern

{

public class ExcelDocument : IDocument

{

public ExcelDocument()

{

Console.WriteLine("Excel Document Created");

}

public void Open()

{

Console.WriteLine("Opening the Excel");

}

public void Close()

{

Console.WriteLine("Closing the Excel");

}

}

}

**PdfDocument.cs**//Submitted by: Superset ID- 6361172

//Exercise 2: Implementing the Factory Method Pattern

using System;

namespace FactoryMethodPattern

{

public class PdfDocument : IDocument

{

public PdfDocument()

{

Console.WriteLine("PDF Document Created");

}

public void Open()

{

Console.WriteLine("Opening the PDF");

}

public void Close()

{

Console.WriteLine("Closing the PDF");

}

}

}

**WordDocument.cs**//Submitted by: Superset ID- 6361172

//Exercise 2: Implementing the Factory Method Pattern

using System;

namespace FactoryMethodPattern

{

public class WordDocument : IDocument

{

public WordDocument()

{

Console.WriteLine("Word Document Created");

}

public void Open()

{

Console.WriteLine("Opening the Word");

}

public void Close()

{

Console.WriteLine("Closing the Word");

}

}

}

**DocumentFactory.cs**

//Submitted by: Superset ID- 6361172

//Exercise 2: Implementing the Factory Method Pattern

namespace FactoryMethodPattern

{

public abstract class DocumentFactory

{

public abstract IDocument CreateDocument();

}

}

**WordDocumentFactory.cs**

//Submitted by: Superset ID- 6361172

//Exercise 2: Implementing the Factory Method Pattern

namespace FactoryMethodPattern

{

public class WordDocumentFactory : DocumentFactory

{

public override IDocument CreateDocument()

{

return new WordDocument();

}

}

}

**PdfDocumentFactory.cs**

//Submitted by: Superset ID- 6361172

//Exercise 2: Implementing the Factory Method Pattern

namespace FactoryMethodPattern

{

public class PdfDocumentFactory : DocumentFactory

{

public override IDocument CreateDocument()

{

return new PdfDocument();

}

}

}

**ExcelDocumentFactory.cs**

//Submitted by: Superset ID- 6361172

//Exercise 2: Implementing the Factory Method Pattern

using FactoryMethodPattern;

namespace FactoryMethodPattern

{

public class ExcelDocumentFactory : DocumentFactory

{

public override IDocument CreateDocument()

{

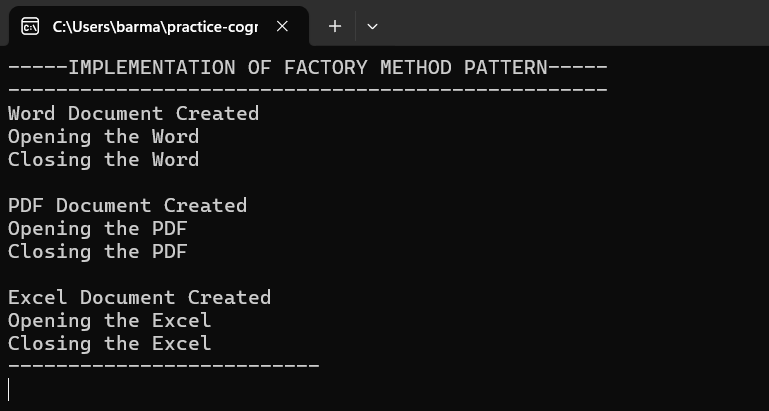
return new ExcelDocument();

}

}

}

**OUTPUT:**

****