**Week 1:**

**Design Pattern and principle:**

**1.Exercise 1:Implementing the Singleton Pattern**

SingletonPatternExample:

[**Logger.cs**](http://logger.cs)

using System;

namespace SingletonPatternExample

{

public sealed class Logger

{

private static readonly Logger \_instance = new Logger();

private Logger() { }

public static Logger Instance => \_instance;

public void Log(string message)

{

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

}

}

}

[**Program.cs**](http://program.cs)

using System;

using SingletonPatternExample;

namespace SingletonPatternExample

{

class Program

{

static void Main(string[] args)

{

Logger logger1 = Logger.Instance;

Logger logger2 = Logger.Instance;

Console.WriteLine($"Are logger1 and logger2 the same instance? {ReferenceEquals(logger1, logger2)}");

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

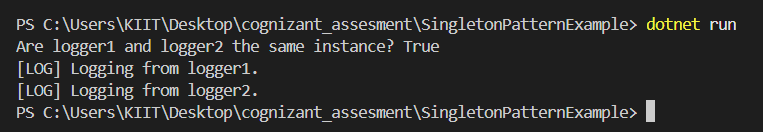
logger2.Log("Logging from logger2.");

}

}

}

**Output:**



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

FactoryMethodExample:

[**Document.cs**](http://document.cs)

**namespace FactoryMethodPattern**

**{**

**public abstract class Document**

**{**

**public abstract void Open();**

**}**

**}**

[**WordDocument.cs**](http://worddocument.cs)

**namespace FactoryMethodPattern**

**{**

**public class WordDocument : Document**

**{**

**public override void Open()**

**{**

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

**}**

**}**

**}**

[**ExcelDocument.cs**](http://exceldocument.cs)

**namespace FactoryMethodPattern**

**{**

**public class ExcelDocument : Document**

**{**

**public override void Open()**

**{**

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

**}**

**}**

**}**

[**PdfDocument.cs**](http://pdfdocument.cs)

**namespace FactoryMethodPattern**

**{**

**public class PdfDocument : Document**

**{**

**public override void Open()**

**{**

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

**}**

**}**

**}**

[**DocumentFactory.cs**](http://documentfactory.cs)

**namespace FactoryMethodPattern**

**{**

**public abstract class DocumentFactory**

**{**

**public abstract Document CreateDocument();**

**}**

**}**

[**WordDocumentFactory.cs**](http://worddocument.cs)

**namespace FactoryMethodPattern**

**{**

**public class WordDocumentFactory : DocumentFactory**

**{**

**public override Document CreateDocument()**

**{**

**return new WordDocument();**

**}**

**}**

**}**

[**ExcelDocumentFactory.cs**](http://exceldocumentfactory.cs)

**namespace FactoryMethodPattern**

**{**

**public class ExcelDocumentFactory : DocumentFactory**

**{**

**public override Document CreateDocument()**

**{**

**return new ExcelDocument();**

**}**

**}**

**}**

[**PdfDocumentFactory.cs**](http://pdfdocumentfactory.cs)

**namespace FactoryMethodPattern**

**{**

**public class PdfDocumentFactory : DocumentFactory**

**{**

**public override Document CreateDocument()**

**{**

**return new PdfDocument();**

**}**

**}**

**}**

[**Program.cs**](http://program.cs)

**using System;**

**namespace FactoryMethodPattern**

**{**

**class Program**

**{**

**static void Main(string[] args)**

**{**

**DocumentFactory wordFactory = new WordDocumentFactory();**

**Document wordDoc = wordFactory.CreateDocument();**

**wordDoc.Open();**

**DocumentFactory pdfFactory = new PdfDocumentFactory();**

**Document pdfDoc = pdfFactory.CreateDocument();**

**pdfDoc.Open();**

**DocumentFactory excelFactory = new ExcelDocumentFactory();**

**Document excelDoc = excelFactory.CreateDocument();**

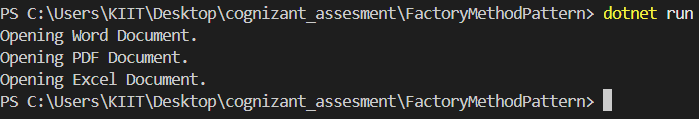
**excelDoc.Open();**

**}**

**}**

**}**

**Output:**

****