**Exercise 1: Implementing the Singleton Pattern**

**Logger class:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace SingletonPatternExample

{

public class Logger

{

private static Logger \_instance;

private Logger()

{

Console.WriteLine("Logger instance created");

}

public static Logger GetInstance()

{

if (\_instance == null)

{

\_instance = new Logger();

}

return \_instance;

}

public void Log(string message)

{

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

}

}

}

**Driver code(Program.cs):**

using SingletonPatternExample;

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");

Console.WriteLine(Object.ReferenceEquals(logger1, logger2)

? "Same instance (Singleton works)"

: "Different instances (Singleton failed)");

}

}

**Output:**



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

**Document Interface:**

using System;

public interface IDocument

{

void Open();

}

**WordDocument.cs:**

using System;

public class WordDocument : IDocument

{

public void Open()

{

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

}

}

**PdfDocument.cs:**

using System;

public class PdfDocument : IDocument

{

public void Open()

{

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

}

}

**ExcelDocument.cs:**

using System;

public class PdfDocument : IDocument

{

public void Open()

{

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

}

}

**WordDocumentFactory.cs:**

public class WordDocumentFactory : DocumentFactory

{

public override IDocument CreateDocument()

{

return new WordDocument();

}

}

**PdfDocumentFactory.cs:**

public class PdfDocumentFactory : DocumentFactory

{

public override IDocument CreateDocument()

{

return new PdfDocument();

}

}

**ExcelDocumentFactory.cs:**

public class ExcelDocumentFactory : DocumentFactory

{

public override IDocument CreateDocument()

{

return new ExcelDocument();

}

}

**DocumentFactory.cs:**

public abstract class DocumentFactory

{

public abstract IDocument CreateDocument();

}

**Driver code(Program.cs):**

using System;

class Program

{

static void Main(string[] args)

{

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();

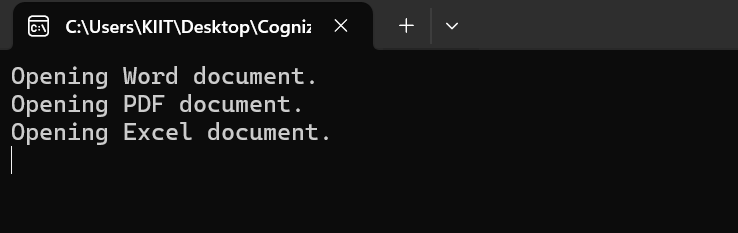
excel.Open();

Console.ReadLine();

}

}

**Output:**

****