**Exercise 1: Implementing the Singleton Pattern**  
  
Solution Code:  
  
public class Logger

{

private static Logger instance;

private Logger()

{

Console.WriteLine("Initialized Logger instance");

}

public static Logger GetInstance()

{

if (instance == null)

{

instance = new Logger();

}

return instance;

}

public void Log(string message)

{

Console.WriteLine("Log Message: " + message);

}

}

class Program

{

static void Main(string[] args)

{

Logger logger1 = Logger.GetInstance();

Logger logger2 = Logger.GetInstance();

logger1.Log("First message");

logger2.Log("Second message");

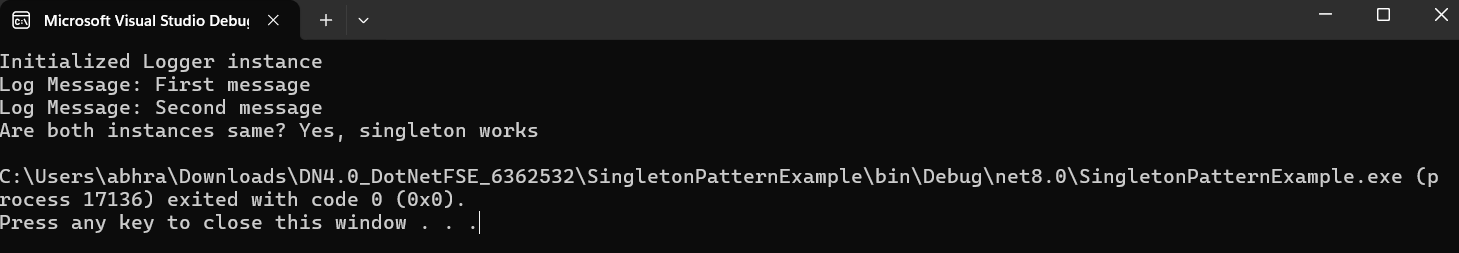
Console.WriteLine("Are both instances same? " +

(logger1 == logger2 ? "Yes, singleton works" : "No, singleton failed"));

}

}

Output:



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

Solution Code:

public interface IDocument

{

void Create();

}

public class WordDocument : IDocument

{

public void Create()

{

Console.WriteLine("Creating Word document");

}

}

public class PdfDocument : IDocument

{

public void Create()

{

Console.WriteLine("Creating PDF document");

}

}

public class ExcelDocument : IDocument

{

public void Create()

{

Console.WriteLine("Creating Excel document");

}

}

public class DocumentFactory

{

public IDocument CreateDocument(string type)

{

switch (type.ToLower())

{

case "word":

return new WordDocument();

case "pdf":

return new PdfDocument();

case "excel":

return new ExcelDocument();

default:

throw new ArgumentException("Unknown document type: " + type);

}

}

}

class Program

{

static void Main(string[] args)

{

DocumentFactory factory = new DocumentFactory();

IDocument word = factory.CreateDocument("word");

word.Create();

IDocument excel = factory.CreateDocument("excel");

excel.Create();

IDocument pdf = factory.CreateDocument("pdf");

pdf.Create();

//Exception handling for unknown document type

IDocument unknown = factory.CreateDocument("unknown");

unknown.Create();

}

}

Output:  
