**Exercise 1: Implementing the Singleton Pattern**

#1 Defining and Implementing LoggerClass

using System;

namespace SingletonPatternExample

{

public class Logger

{

private static Logger instance;

private static readonly object lockObj = new object();

private Logger()

{

Console.WriteLine("Logger initialized.");

}

public static Logger GetInstance()

{

if (instance == null)

{

lock (lockObj)

{

if (instance == null)

{

instance = new Logger();

}

}

}

return instance;

}

public void Log(string message)

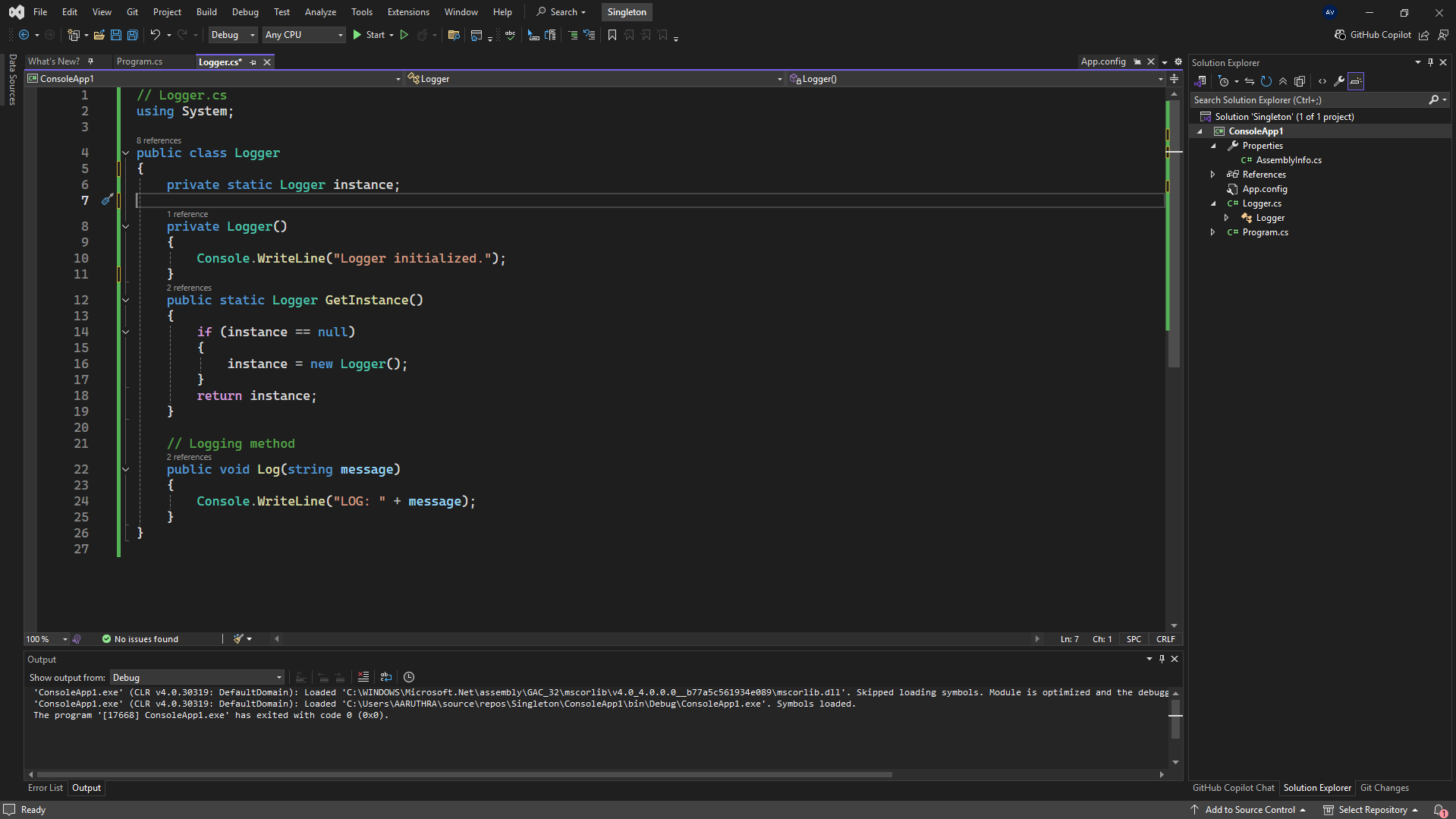
{

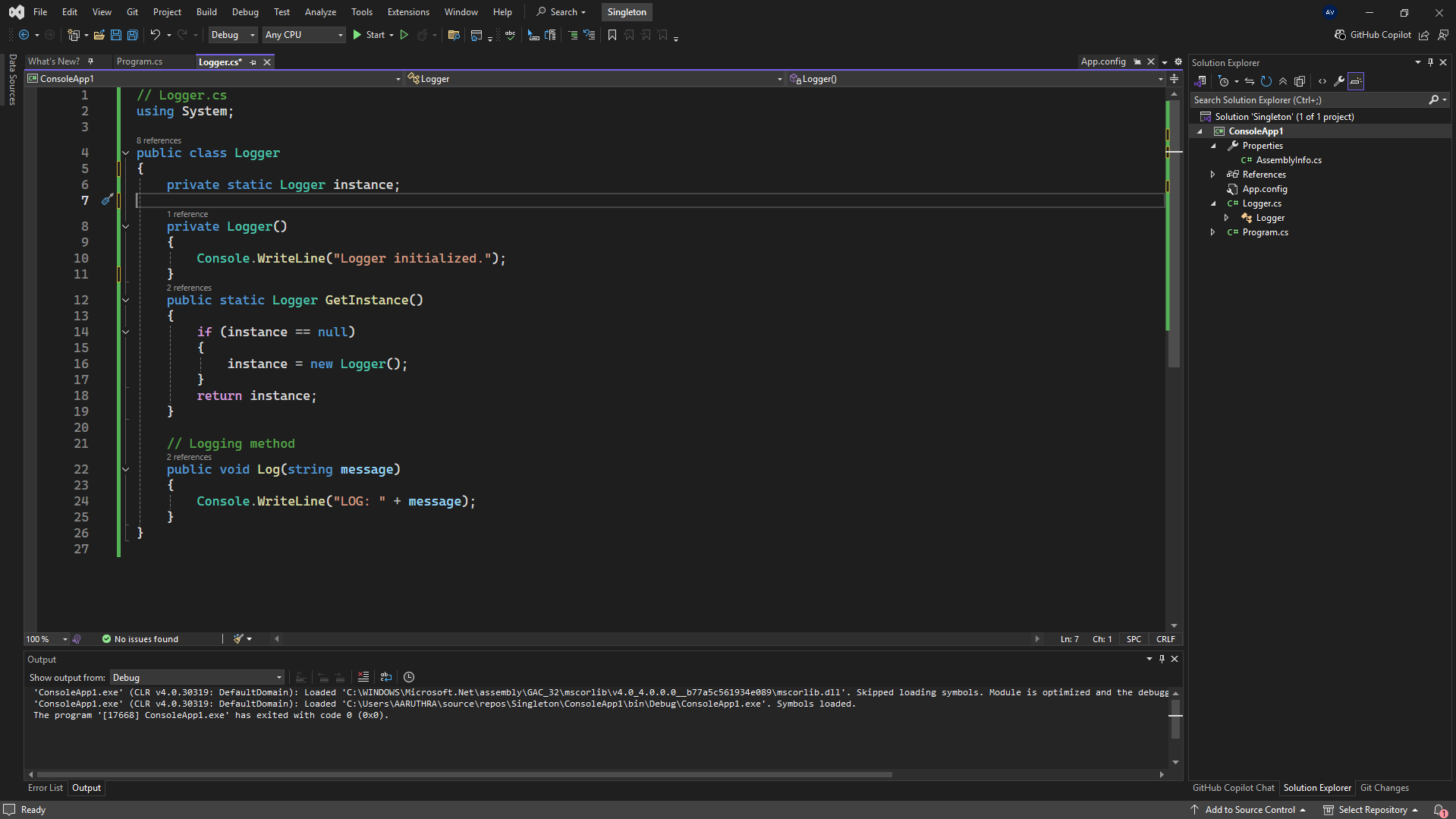
Console.WriteLine("LOG: " + message);

}

}

}





#2 Testing SingletonClass

using System;

namespace SingletonPatternExample

{

class Program

{

static void Main(string[] args)

{

Logger logger1 = Logger.GetInstance();

Logger logger2 = Logger.GetInstance();

logger1.Log("First log message");

logger2.Log("Second log message");

if (ReferenceEquals(logger1, logger2))

{

Console.WriteLine("Both logger instances are the same.");

}

else

{

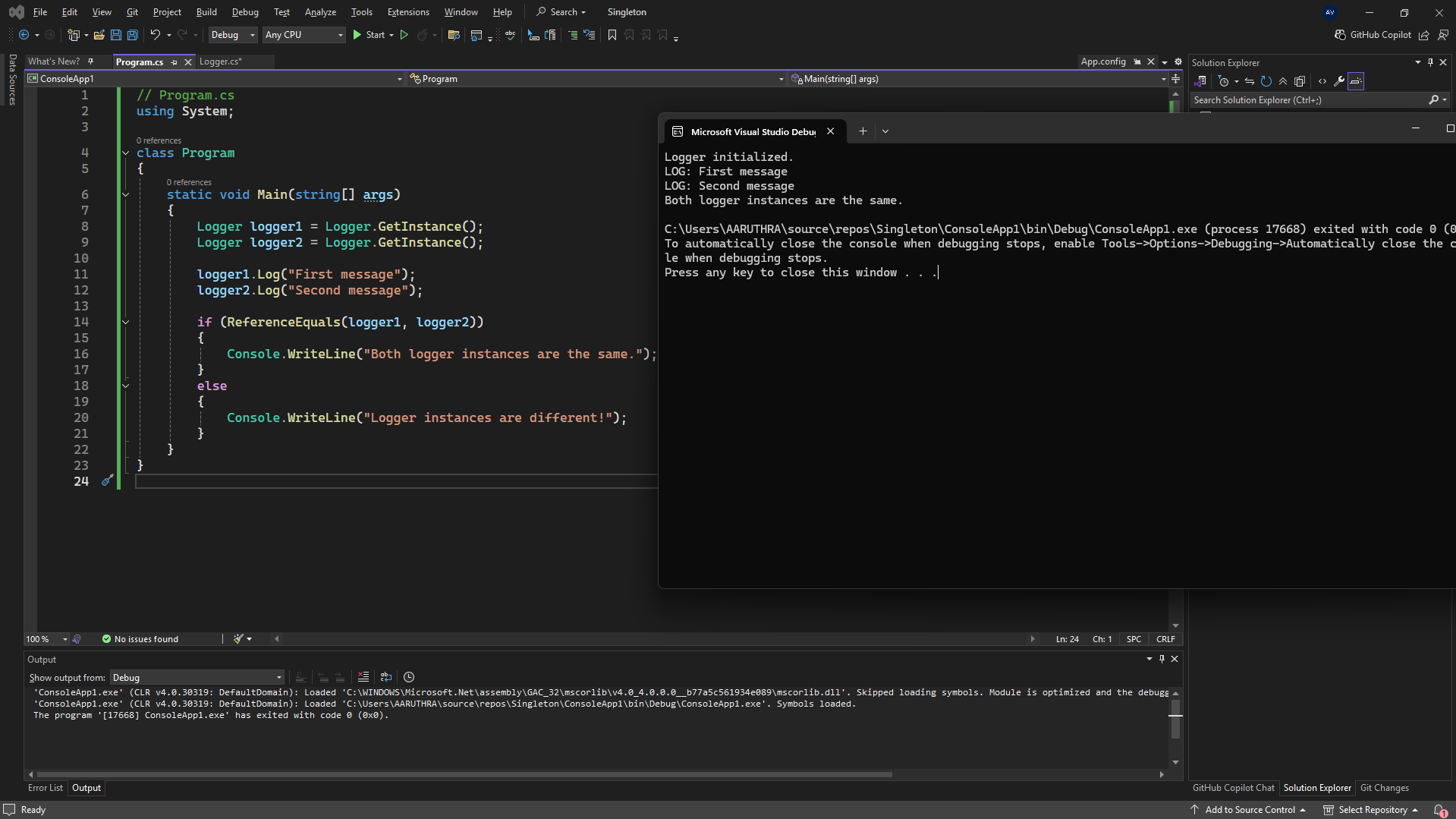
Console.WriteLine("Logger instances are different!");

}

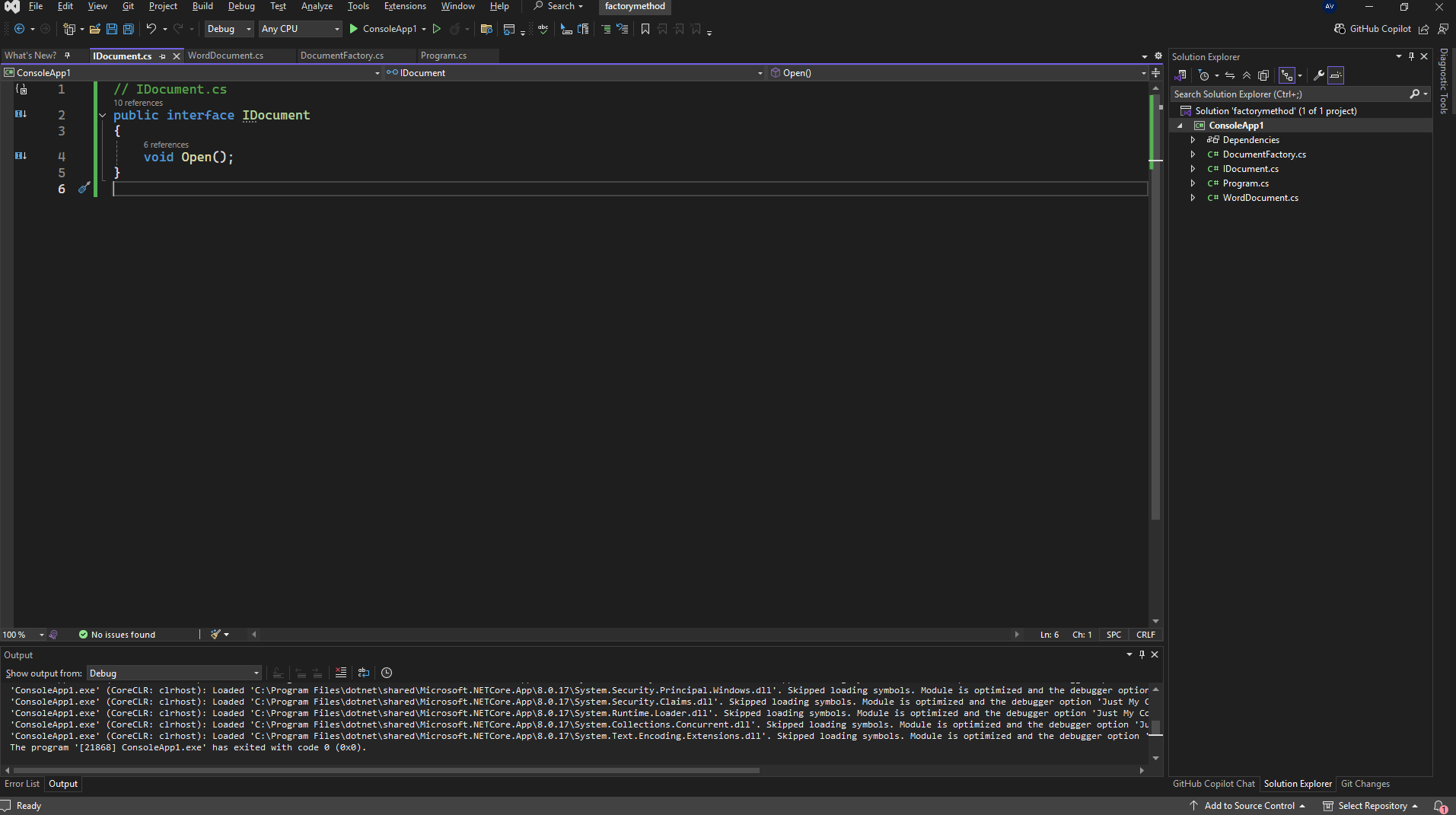
}

}

}

#3 Output

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

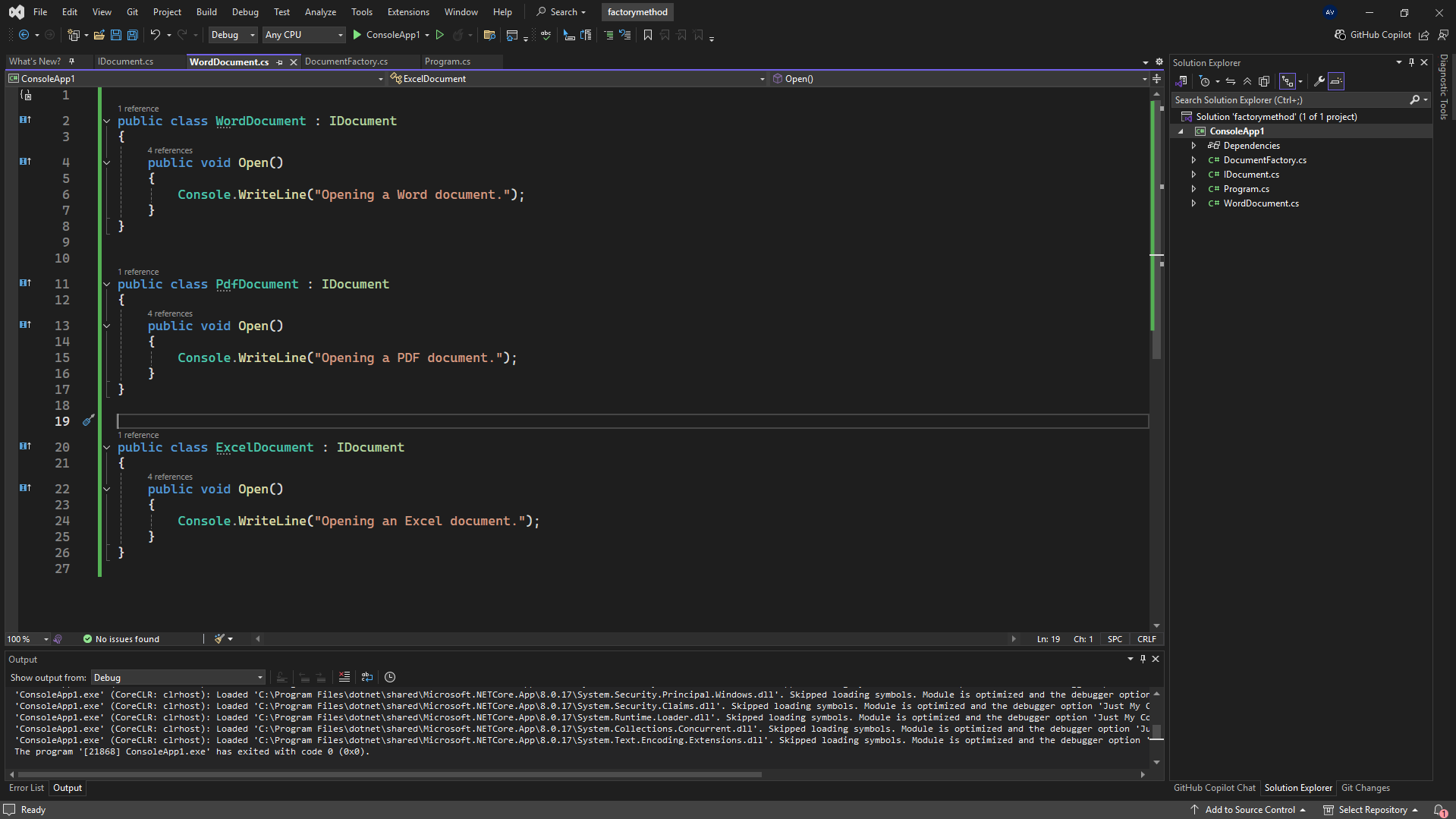
**#1 Defining Document**

public interface IDocument

{

void Open();

}

#2 Document Creation

// WordDocument.cs

public class WordDocument : IDocument

{

public void Open()

{

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

}

}

// PdfDocument.cs

public class PdfDocument : IDocument

{

public void Open()

{

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

}

}

// ExcelDocument.cs

public class ExcelDocument : IDocument

{

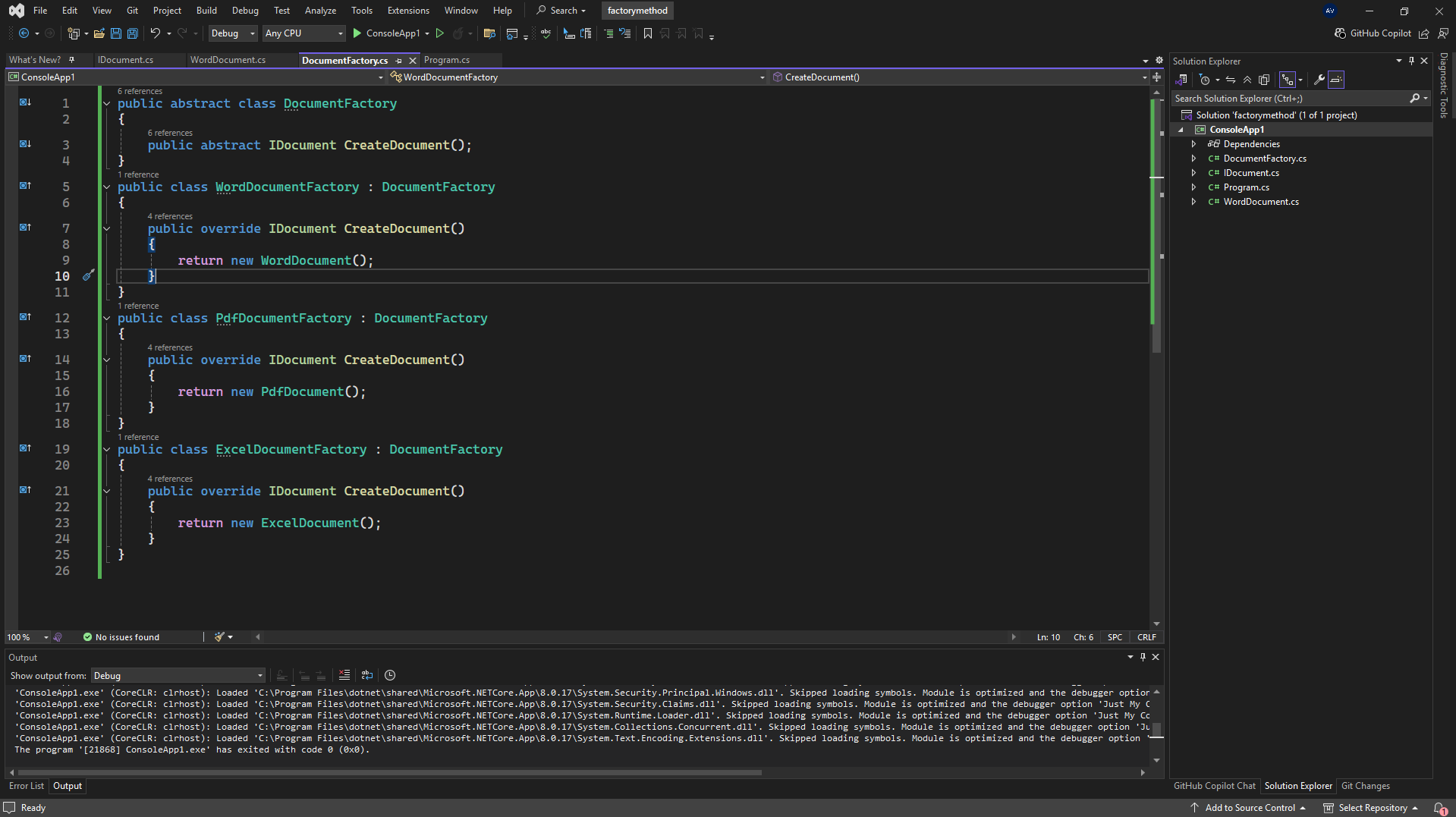
public void Open()

{

Console.WriteLine("Opening an Excel document.");

}

}

#3 Factory Method Implementation

// DocumentFactory.cs

public abstract class DocumentFactory

{

public abstract IDocument CreateDocument();

}

// 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

{

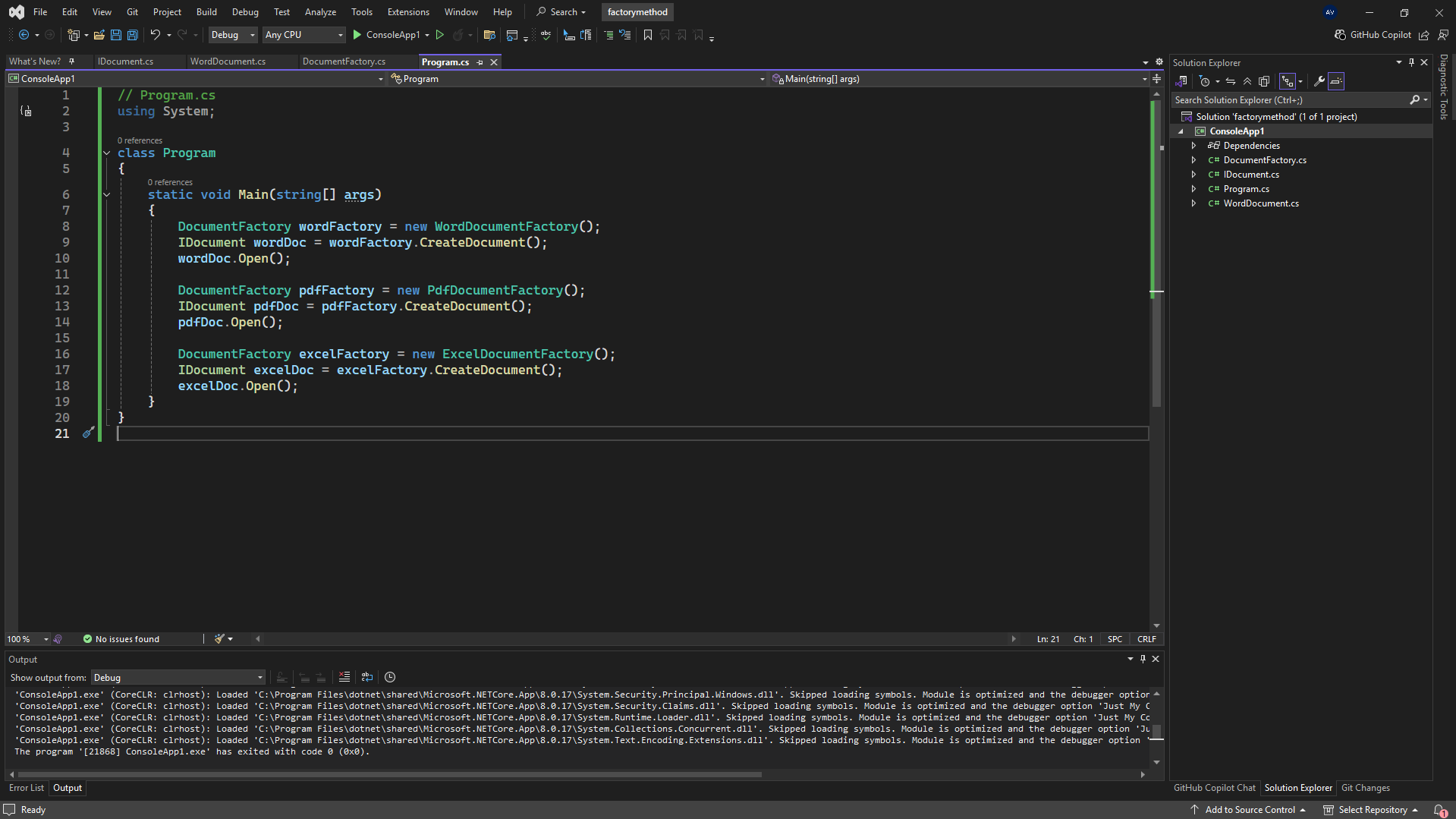
public override IDocument CreateDocument()

{

return new ExcelDocument();

}

}

#4 Testing Factory Method

// Program.cs

using System;

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();

}

}