**Design Patterns and Principles**

Exercise 2: Implementing the Factory Method Pattern

**CODE:**

public interface IDocument

{

void Open();

}

public class WordDocument : IDocument

{

public void Open()

{

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

}

}

public class PdfDocument : IDocument

{

public void Open()

{

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

}

}

public class ExcelDocument : IDocument

{

public void Open()

{

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

}

}

public abstract class DocumentFactory

{

public abstract IDocument CreateDocument();

}

public class WordDocumentFactory : DocumentFactory

{

public override IDocument CreateDocument()

{

return new WordDocument();

}

}

public class PdfDocumentFactory : DocumentFactory

{

public override IDocument CreateDocument()

{

return new PdfDocument();

}

}

public class ExcelDocumentFactory : DocumentFactory

{

public override IDocument CreateDocument()

{

return new ExcelDocument();

}

}

* 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();

Console.ReadLine();

}

}

* OUTPUT

