

Exercise 1: Implementing the Singleton Pattern

Code:

Program.cs

```
...  
  
using System;  
  
public class Program  
{  
    public static void Main(string[] args)  
    {  
        Logger log1 = Logger.GetInstance();  
        log1.Print("This is the First message");  
  
        Logger log2 = Logger.GetInstance();  
        log2.Print("This is the Second Message");  
    }  
}  
...
```

Logger.cs

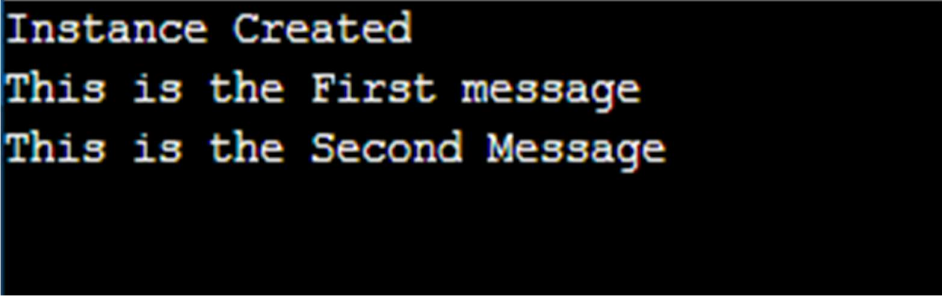
```
...  
  
using System;  
  
public class Logger  
{  
    private static Logger instance = null;  
    private Logger()  
    {  
        Console.WriteLine("Instance Created");  
    }  
    public static Logger GetInstance()  
    {  
        if (instance == null)  
        {
```

```

instance = new Logger();
}
return instance;
}
public void Print(string message)
{
    Console.WriteLine(message);
}
}
...

```

OUTPUT :



```

Instance Created
This is the First message
This is the Second Message

```

Exercise 2: Implementing the Factory Method Pattern

IDocument.cs :

```

public interface IDocument
{
    void Open();
}

```

WordDocument.cs:

```

using System;

public class WordDocument : IDocument
{
    public void Open()
    {
        Console.WriteLine("Opening a Word Document.");
    }
}

```

```
}
```

```
}
```

PdfDocument.cs:

```
using System;
```

```
public class PdfDocument : IDocument
```

```
{
```

```
    public void Open()
```

```
    {
```

```
        Console.WriteLine("Opening a PDF Document.");
```

```
    }
```

```
}
```

ExcelDocument.cs:

```
using System;
```

```
public class ExcelDocument : IDocument
```

```
{
```

```
    public void Open()
```

```
    {
```

```
        Console.WriteLine("Opening an Excel Document.");
```

```
    }
```

```
}
```

DocumentFactory.cs:

```
public abstract class DocumentFactory
```

```
{
```

```
    public abstract IDocument CreateDocument();
```

```
}
```

WordFactory.cs:

```
public class WordFactory : DocumentFactory
```

```
{
```

```
    public override IDocument CreateDocument()
```

```
    {
```

```
        return new WordDocument();
```

```
}
```

```
}
```

PdfFactory.cs:

```
public class PdfFactory : DocumentFactory
{
    public override IDocument CreateDocument()
    {
        return new PdfDocument();
    }
}
```

ExcelFactory.cs:

```
public class ExcelFactory : DocumentFactory
{
    public override IDocument CreateDocument()
    {
        return new ExcelDocument();
    }
}
```

Program.cs:

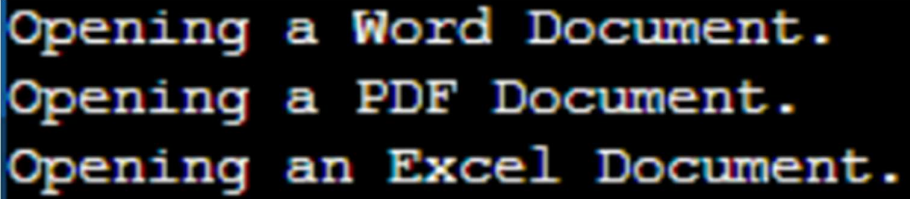
```
using System;

public class Program
{
    public static void Main(string[] args)
    {
        DocumentFactory wordFactory = new WordFactory();
        IDocument wordDoc = wordFactory.CreateDocument();
        wordDoc.Open();

        DocumentFactory pdfFactory = new PdfFactory();
        IDocument pdfDoc = pdfFactory.CreateDocument();
        pdfDoc.Open();

        DocumentFactory excelFactory = new ExcelFactory();
```

```
IDocument excelDoc = excelFactory.CreateDocument();  
excelDoc.Open();  
}  
}
```



```
Opening a Word Document.  
Opening a PDF Document.  
Opening an Excel Document.
```