**Exercise 1: Implementing the Singleton Pattern**

**LoggerService.java:**public class LoggerService

{

    private static LoggerService instance;

    private LoggerService()

    {

        System.out.println("Logger initialized");

    }

    public static LoggerService getInstance()

    {

        if (instance==null)

            instance=new LoggerService();

        return instance;

    }

    public void log(String message)

    {

        System.out.println("LOG: "+message);

    }

}

**AppLoggerDemo.java:**public class AppLoggerDemo

{

    public static void main(String[] args)

    {

        LoggerService loggerA=LoggerService.getInstance();

        loggerA.log("App has started");

        LoggerService loggerB=LoggerService.getInstance();

        loggerB.log("App is running");

        if (loggerA==loggerB)

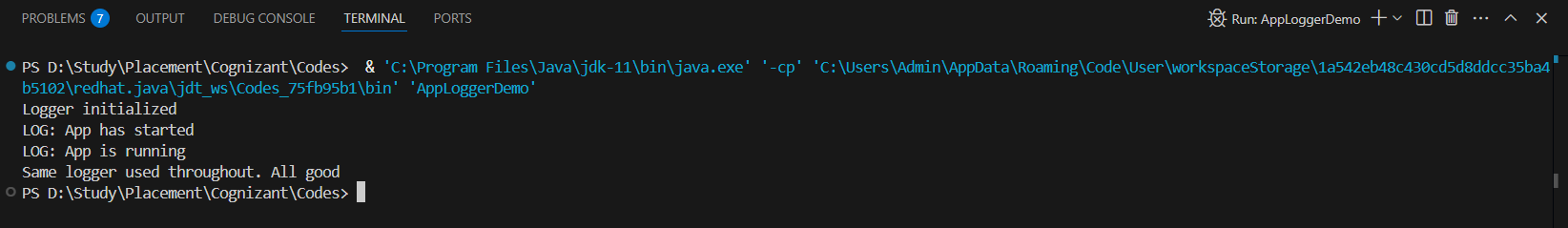
            System.out.println("Same logger used throughout. All good");

        else

            System.out.println("Different loggers found. Something wrong.");

    }

}

**Output:**

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

**AppDocumentDemo.class:**public class AppDocumentDemo {

   public AppDocumentDemo() {

   }

   public static void main(String[] var0) {

      WordFactory var1 = new WordFactory();

      Document var2 = var1.createDocument();

      var2.open();

      PdfFactory var3 = new PdfFactory();

      Document var4 = var3.createDocument();

      var4.open();

      ExcelFactory var5 = new ExcelFactory();

      Document var6 = var5.createDocument();

      var6.open();

   }

}

**Document.class:**  
public interface Document {

   void open();

}

**DocumentFactory.class:**  
public abstract class DocumentFactory {

   public DocumentFactory() {

   }

   public abstract Document createDocument();

}

**ExcelDoc.class:**  
public class ExcelDoc implements Document {

   public ExcelDoc() {

   }

   public void open() {

      System.out.println("Opening an Excel document.");

   }

}

**ExcelFactory.class:**  
public class ExcelFactory extends DocumentFactory {

   public ExcelFactory() {

   }

   public Document createDocument() {

      return new ExcelDoc();

   }

}

**PdfDoc.class:**  
public class PdfDoc implements Document {

   public PdfDoc() {

   }

   public void open() {

      System.out.println("Opening a PDF document.");

   }

}

**PdfFactoryClass:**  
public class PdfFactory extends DocumentFactory {

   public PdfFactory() {

   }

   public Document createDocument() {

      return new PdfDoc();

   }

}

**WordDoc.class:**  
public class WordDoc implements Document {

   public WordDoc() {

   }

   public void open() {

      System.out.println("Opening a Word document.");

   }

}

**WordFactory.class:**  
public class WordFactory extends DocumentFactory {

   public WordFactory() {

   }

   public Document createDocument() {

      return new WordDoc();

   }

}

**Output:**

