

DESIGN PATTERNS AND PRINCIPLES

MANDATORY HANDS-ON

Exercise 1: Implementing the Singleton Pattern

Logger.java

```
public class Logger {  
    private static Logger instance;  
  
    private Logger() {  
        System.out.println("Logger initialized.");  
    }  
  
    public static Logger getInstance() {  
        if (instance == null) {  
            instance = new Logger();  
        }  
        return instance;  
    }  
  
    public void log(String message) {  
        System.out.println("[LOG]: " + message);  
    }  
}
```

Main.java

```
public class Main {  
    public static void main(String[] args) {  
        Logger logger1 = Logger.getInstance();  
        logger1.log("Application started.");  
        Logger logger2 = Logger.getInstance();  
        logger2.log("User logged in.");  
        if (logger1 == logger2) {  
            System.out.println("logger1 and logger2 refer to the same instance.");  
        }  
    }  
}
```

```

    }
else {
    System.out.println("logger1 and logger2 refer to different instances.");
}
}
}
}

```

OUTPUT:

```

7 *****
8 public class Main {
9     public static void main(String[] args) {
10         Logger logger1 = Logger.getInstance();
11         logger1.log("Application started.");
12
13         Logger logger2 = Logger.getInstance();
14         logger2.log("User logged in.");
15
16         if (logger1 == logger2) {
17             System.out.println("logger1 and logger2 refer to the same instance.");
18         } else {
19             System.out.println("logger1 and logger2 refer to different instances.");
20         }
21     }
22 }
23
24

```

Logger initialized.
[LOG]: Application started.
[LOG]: User logged in.
logger1 and logger2 refer to the same instance.
...Program finished with exit code 0
Press ENTER to exit console.

Logger initialized.

[LOG]: Application started.

[LOG]: User logged in.

logger1 and logger2 refer to the same instance.

Exercise 2: Implementing the Factory Method Pattern

FactoryPatternMethod.java

```

interface Document {
    void open();
}

class WordDocument implements Document {

```

```

    public void open() {
        System.out.println("Opening a Word document.");
    }
}

class PdfDocument implements Document {
    public void open() {
        System.out.println("Opening a PDF document.");
    }
}

class ExcelDocument implements Document {
    public void open() {
        System.out.println("Opening an Excel document.");
    }
}

abstract class DocumentFactory {
    public abstract Document createDocument();
}

class WordDocumentFactory extends DocumentFactory {
    public Document createDocument() {
        return new WordDocument();
    }
}

class PdfDocumentFactory extends DocumentFactory {
    public Document createDocument() {
        return new PdfDocument();
    }
}

class ExcelDocumentFactory extends DocumentFactory {
    public Document createDocument() {
        return new ExcelDocument();
    }
}

```

```

    }
}

public class FactoryMethodPatternExample {
    public static void main(String[] args) {
        DocumentFactory wordFactory = new WordDocumentFactory();
        Document word = wordFactory.createDocument();
        word.open();

        DocumentFactory pdfFactory = new PdfDocumentFactory();
        Document pdf = pdfFactory.createDocument();
        pdf.open();

        DocumentFactory excelFactory = new ExcelDocumentFactory();
        Document excel = excelFactory.createDocument();
        excel.open();
    }
}

```

OUTPUT:

```

8 interface Document {
9     void open();
10 }
11
12 class WordDocument implements Document {
13     public void open() {
14         System.out.println("Opening a Word document.");
15     }
16 }
17
18 class PdfDocument implements Document {
19     public void open() {
20         System.out.println("Opening a PDF document.");
21     }
22 }
23
24 class ExcelDocument implements Document {
25     public void open() {
26         System.out.println("Opening an Excel document.");
27     }
28 }
29
30 public class FactoryMethodPatternExample {
31     public static void main(String[] args) {
32         DocumentFactory wordFactory = new WordDocumentFactory();
33         Document word = wordFactory.createDocument();
34         word.open();
35
36         DocumentFactory pdfFactory = new PdfDocumentFactory();
37         Document pdf = pdfFactory.createDocument();
38         pdf.open();
39
40         DocumentFactory excelFactory = new ExcelDocumentFactory();
41         Document excel = excelFactory.createDocument();
42         excel.open();
43     }
44 }

```

Opening a Word document.
Opening a PDF document.
Opening an Excel document.
...Program finished with exit code 0
Press ENTER to exit console.

Opening a Word document.

Opening a PDF document.

Opening an Excel document.