Digital Nurture 4.0 – Deep Skilling

Week 1 - Mandatory Hands - on

Hands - on : 1

1) Implementing the Singleton Pattern

Logger.java

```
package SingletonPatternExample;

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);
    }
}
```

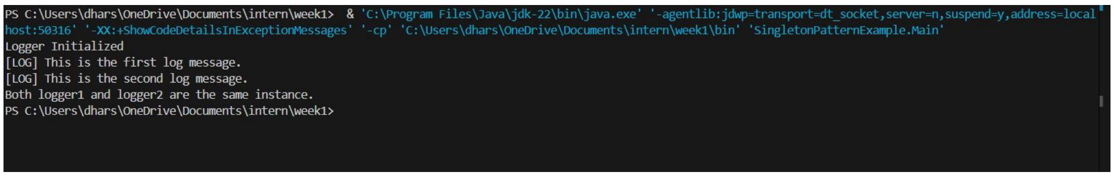
Main.java

```
package SingletonPatternExample;

public class Main {
    public static void main(String[] args) {
        Logger logger1 = Logger.getInstance();
        logger1.log("This is the first log message.");

        Logger logger2 = Logger.getInstance();
        logger2.log("This is the second log message.");

        if (logger1 == logger2) {
            System.out.println("Both logger1 and logger2 are the same instance.");
        } else {
            System.out.println("Different instances exist! Singleton failed.");
        }
    }
}
```

App.java

```
package SingletonPatternExample;

public class App {
    public static void main(String[] args) {
        Logger logger1 = Logger.getInstance();
        logger1.log("Logging from logger1");

        Logger logger2 = Logger.getInstance();
        logger2.log("Logging from logger2");

        if (logger1 == logger2) {
            System.out.println("Singleton confirmed: Both logger1 and logger2 are the
same instance.");
        } else {
            System.out.println("Singleton failed: Different instances exist.");
        }
    }
}
```

Output



```
PS C:\Users\dhars\OneDrive\Documents\intern\week1>  & 'C:\Program Files\Java\jdk-22\bin\java.exe' '-agentlib:jdwp=transport=dt_socket,server=n,suspend=y,address=local
host:50316' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\dhars\OneDrive\Documents\intern\week1\bin' 'SingletonPatternExample.Main'
Logger Initialized
[LOG] This is the first log message.
[LOG] This is the second log message.
Both logger1 and logger2 are the same instance.
PS C:\Users\dhars\OneDrive\Documents\intern\week1>
```

Hands - on : 2

2) Implementing the Factory Method Pattern

Document.java

```
package FactoryMethodPatternExample;

public interface Document {
    void open();
}
```

DocumentFactory.java

```
package FactoryMethodPatternExample;

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

ExcelDocument.java

```
package FactoryMethodPatternExample;

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

ExcelDocumentFactory.java

```
package FactoryMethodPatternExample;

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

PdfDocument.java

```
package FactoryMethodPatternExample;

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

pdfDocumentFactory.java

```
package FactoryMethodPatternExample;

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

WordDocument.java

```
package FactoryMethodPatternExample;

public class WordDocument implements Document {
   @Override
   public void open() {
      System.out.println("Opening Word document...");
   }
}
```

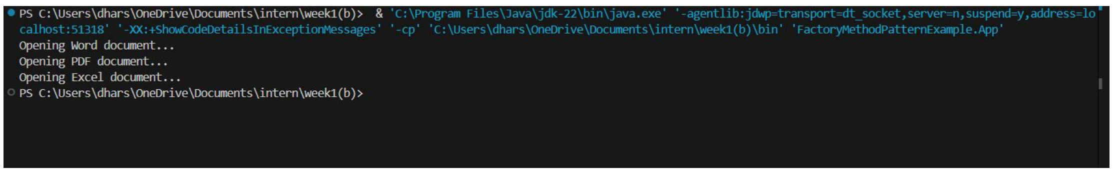
WordDocumentFactory.java

```
package FactoryMethodPatternExample;

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

App.java

```java
package FactoryMethodPatternExample;

public class App {
    public static void main(String[] args) {

        DocumentFactory wordFactory = new WordDocumentFactory();
        Document wordDoc = wordFactory.createDocument();
        wordDoc.open();

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

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

Output



```
PS C:\Users\dhars\OneDrive\Documents\intern\week1(b)>  & 'C:\Program Files\Java\jdk-22\bin\java.exe' '-agentlib:jdwp=transport=dt_socket,server=n,suspend=y,address=lo
calhost:51318' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\dhars\OneDrive\Documents\intern\week1(b)\bin' 'FactoryMethodPatternExample.App'
Opening Word document...
Opening PDF document...
Opening Excel document...
PS C:\Users\dhars\OneDrive\Documents\intern\week1(b)>
```

Hands - on : 3

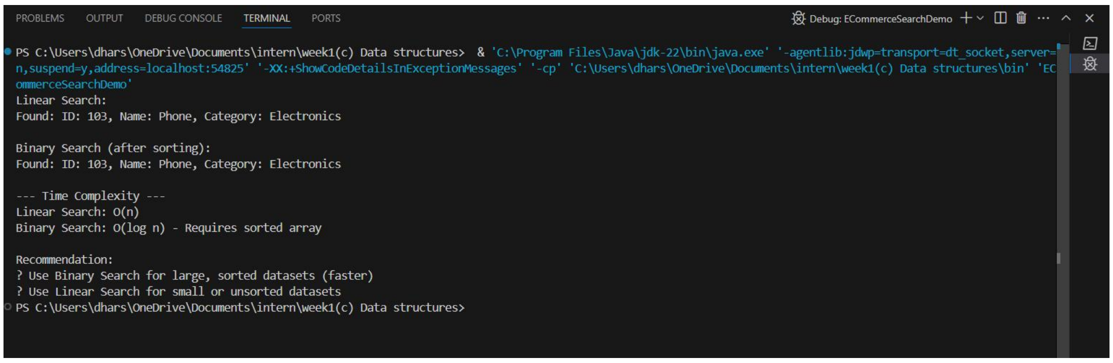
3) E-commerce Platform Search Function

ECommerceSearchDemo.java

```
import java.util.Arrays;

class Product implements Comparable<Product> {
    int productId;
    String productName;
    String category;

    public Product(int productId, String productName, String category) {
        this.productId = productId;
        this.productName = productName;
        this.category = category;
    }

    @Override
    public int compareTo(Product other) {
        return this.productName.compareToIgnoreCase(other.productName);
    }

    @Override
    public String toString() {
        return "ID: " + productId + ", Name: " + productName + ", Category: " + category;
    }
}

class SearchFunctions {
    public static Product linearSearch(Product[] products, String targetName) {
        for (Product p : products) {
            if (p.productName.equalsIgnoreCase(targetName)) {
                return p;
            }
        }
        return null;
    }

    public static Product binarySearch(Product[] products, String targetName) {
        int left = 0;
        int right = products.length - 1;

        while (left <= right) {
            int mid = left + (right - left) / 2;
            int comparison =
products[mid].productName.compareToIgnoreCase(targetName);
```

```
            if (comparison == 0)
                return products[mid];
            else if (comparison < 0)
                left = mid + 1;
            else
                right = mid - 1;
        }

        return null;
    }
}

public class ECommerceSearchDemo {
    public static void main(String[] args) {
        Product[] products = {
                new Product(101, "Laptop", "Electronics"),
                new Product(102, "Shoes", "Fashion"),
                new Product(103, "Phone", "Electronics"),
                new Product(104, "Watch", "Accessories"),
                new Product(105, "Bag", "Fashion")
        };

        String searchTarget = "Phone";

        System.out.println("Linear Search:");
        Product found = SearchFunctions.linearSearch(products, searchTarget);
        System.out.println(found != null ? "Found: " + found : "Product not found");

        Arrays.sort(products);

        System.out.println("\nBinary Search (after sorting):");
        Product foundBinary = SearchFunctions.binarySearch(products, searchTarget);
        System.out.println(foundBinary != null ? "Found: " + foundBinary : "Product not
found");

        System.out.println("\n--- Time Complexity ---");
        System.out.println("Linear Search: O(n)");
        System.out.println("Binary Search: O(log n) - Requires sorted array");

        System.out.println("\nRecommendation:");
        System.out.println("→ Use Binary Search for large, sorted datasets (faster)");
        System.out.println("→ Use Linear Search for small or unsorted datasets");
    }
}
```

Output



```
PS C:\Users\dhars\OneDrive\Documents\intern\week1(c) Data structures>  & 'C:\Program Files\Java\jdk-22\bin\java.exe' '-agentlib:jdwp=transport=dt_socket,server=
n,suspend=y,address=localhost:54825' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\dhars\OneDrive\Documents\intern\week1(c) Data structures\bin' 'EC
ommerceSearchDemo'
Linear Search:
Found: ID: 103, Name: Phone, Category: Electronics

Binary Search (after sorting):
Found: ID: 103, Name: Phone, Category: Electronics

--- Time Complexity ---
Linear Search: O(n)
Binary Search: O(log n) - Requires sorted array

Recommendation:
? Use Binary Search for large, sorted datasets (faster)
? Use Linear Search for small or unsorted datasets
PS C:\Users\dhars\OneDrive\Documents\intern\week1(c) Data structures>
```

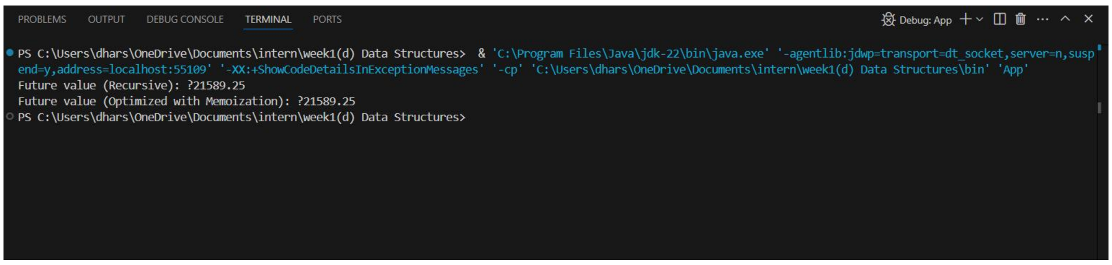
Hands - on : 4

4) Financial Forecasting

App.java

```
public class App {

   public static double forecastRecursive(double baseValue, double growthRate, int
years) {
      if (years == 0) {
         return baseValue;
      }
      return forecastRecursive(baseValue, growthRate, years - 1) * (1 + growthRate);
   }

   public static double forecastMemo(double baseValue, double growthRate, int years,
double[] memo) {
      if (years == 0)
         return baseValue;
      if (memo[years] != 0)
         return memo[years];

      memo[years] = forecastMemo(baseValue, growthRate, years - 1, memo) * (1 +
growthRate);
      return memo[years];
   }

   public static void main(String[] args) {
      double baseValue = 10000;
      double growthRate = 0.08;
      int years = 10;

      double recursiveResult = forecastRecursive(baseValue, growthRate, years);
      System.out.printf("Future value (Recursive): ₹%.2f%n", recursiveResult);

      double[] memo = new double[years + 1];
      double memoResult = forecastMemo(baseValue, growthRate, years, memo);
      System.out.printf("Future value (Optimized with Memoization): ₹%.2f%n",
memoResult);
   }
}
```

Output



```
PS C:\Users\dhars\OneDrive\Documents\intern\week1(d) Data Structures>  & 'C:\Program Files\Java\jdk-22\bin\java.exe' '-agentlib:jdwp=transport=dt_socket,server=n,susp
end=y,address=localhost:55109' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\dhars\OneDrive\Documents\intern\week1(d) Data Structures\bin' 'App'
Future value (Recursive): ?21589.25
Future value (Optimized with Memoization): ?21589.25
PS C:\Users\dhars\OneDrive\Documents\intern\week1(d) Data Structures>
```