

Name: Sangamithra U

SupersetID: 6372485

Data Structures and Algorithms (DSA)

Exercise : Financial Forecasting

This program uses a queue to apply the sliding window technique for moving average calculation and detects trends in historical data to adjust the forecast dynamically. It integrates multiple DSA concepts including queue, arrays, and algorithmic decision-making.

Code:

Forecast.java

```
import java.util.*;

public class Forecast {

    public static void main(String[] args) {

        int[] revenue = {10000, 12000, 13000, 11000, 12500, 14000}; // Monthly revenues

        int windowSize = 3; // We'll use the last 3 months to predict the next

        Queue<Integer> window = new LinkedList<>();

        int sum = 0;

        // Fill the initial window
        for (int i = revenue.length - windowSize; i < revenue.length; i++) {

            window.add(revenue[i]);

            sum += revenue[i];

        }

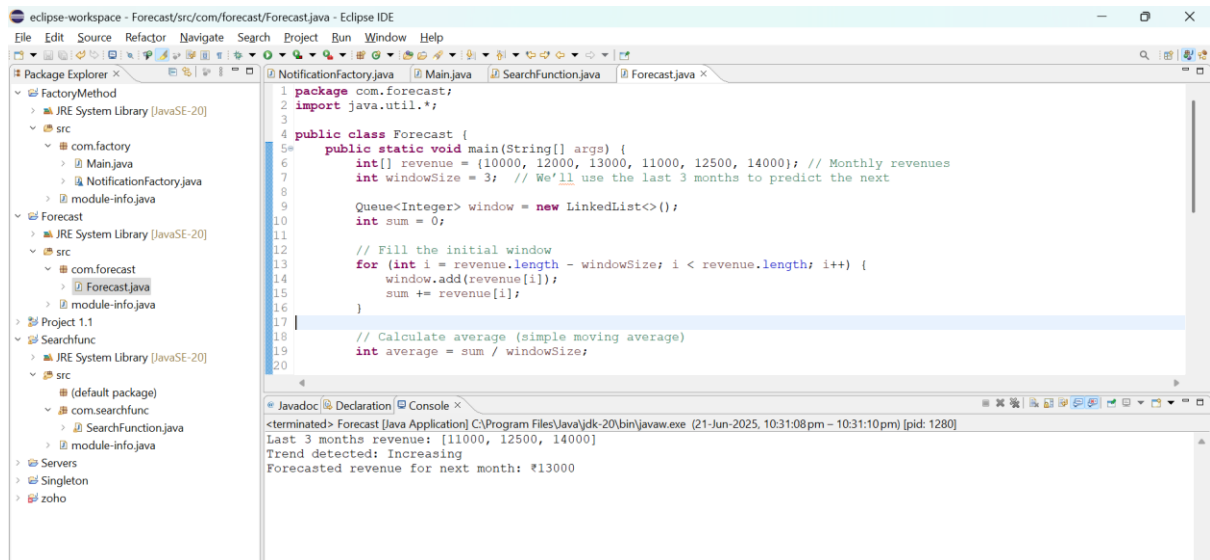
        // Calculate average (simple moving average)
        int average = sum / windowSize;

        // Detect trend direction: Upward, Downward, or Stable
        boolean increasing = true, decreasing = true;

        int[] last = window.stream().mapToInt(Integer::intValue).toArray();
```

```
for (int i = 1; i < last.length; i++) {  
    if (last[i] > last[i - 1]) decreasing = false;  
    if (last[i] < last[i - 1]) increasing = false;  
}  
  
String trend = increasing ? "Increasing" : decreasing ? "Decreasing" : "Stable";  
  
// Project next month revenue based on trend  
int forecast;  
if (increasing) {  
    forecast = average + 500;  
} else if (decreasing) {  
    forecast = average - 500;  
} else {  
    forecast = average;  
}  
  
System.out.println("Last 3 months revenue: " + Arrays.toString(last));  
System.out.println("Trend detected: " + trend);  
System.out.println("Forecasted revenue for next month: ₹" + forecast);  
}  
}
```

Output:



The screenshot displays the Eclipse IDE interface. The Package Explorer on the left shows a project structure with a package named 'com.forecast' containing 'Forecast.java'. The main editor window shows the code for 'Forecast.java', which implements a simple moving average algorithm. The console at the bottom shows the output of the program.

```
1 package com.forecast;
2 import java.util.*;
3
4 public class Forecast {
5     public static void main(String[] args) {
6         int[] revenue = {10000, 12000, 13000, 11000, 12500, 14000}; // Monthly revenues
7         int windowSize = 3; // We'll use the last 3 months to predict the next
8
9         Queue<Integer> window = new LinkedList<>();
10        int sum = 0;
11
12        // Fill the initial window
13        for (int i = revenue.length - windowSize; i < revenue.length; i++) {
14            window.add(revenue[i]);
15            sum += revenue[i];
16        }
17
18        // Calculate average (simple moving average)
19        int average = sum / windowSize;
20    }
}
```

Console Output:

```
<terminated> Forecast [Java Application] C:\Program Files\Java\jdk-20\bin\javaw.exe (21-Jun-2025, 10:31:08 pm - 10:31:10 pm) [pid: 1280]
Last 3 months revenue: [11000, 12500, 14000]
Trend detected: Increasing
Forecasted revenue for next month: ₹13000
```