**Exercise 7: Financial Forecasting**

**Scenario:**

You are developing a financial forecasting tool that predicts future values based on past data.

**Steps:**

1. **Understand Recursive Algorithms:**
   1. Explain the concept of recursion and how it can simplify certain problems.
2. **Setup:**
   1. Create a method to calculate the future value using a recursive approach.
3. **Implementation:**
   1. Implement a recursive algorithm to predict future values based on past growth rates.
4. **Analysis:**
   1. Discuss the time complexity of your recursive algorithm.
   2. Explain how to optimize the recursive solution to avoid excessive computation.

**SOLUTION**

**1. Understand Recursive Algorithms**

**What is Recursion?**Recursion is a method where a function calls itself to solve a smaller version of the same problem**.**

**Why use it?**

* It simplifies problems like factorials, Fibonacci, tree traversal, etc.
* In forecasting, it can help break down multi-step predictions.

**Example analogy:**To calculate future value after n years:

* Future value at year n = value at year n-1 × (1 + growth rate)

**2. Setup – Recursive Forecast Method**

Let’s assume:

* initialValue: value at year 0
* growthRate: annual growth rate (e.g., 10% = 0.10)
* years: number of years to forecast

public class Main {

// Step 2: Recursive method to calculate future value

public static double forecast(double initialValue, double growthRate, int years) {

// Base case: year 0

if (years == 0) {

return initialValue;

}

// Recursive step: future value = previous year's value × (1 + growthRate)

return forecast(initialValue, growthRate, years - 1) \* (1 + growthRate);

}

public static void main(String[] args) {

double initialValue = 1000.0;

double growthRate = 0.10; // 10%

int years = 5;

double futureValue = forecast(initialValue, growthRate, years);

System.out.printf("Forecasted value after %d years: %.2f\n", years, futureValue);

}

}

**3. Implementation: Recursive Forecast**

Example run:

* Initial value = 1000
* Growth rate = 10%
* Years = 5

Output:

Forecasted value after 5 years: 1610.51

A screenshot of a computer

AI-generated content may be incorrect.

**4. Analysis**

**Time Complexity:**

The function calls itself once per year, so:

T(n) = T(n - 1) + O(1) → O(n)

public class Main {

// Recursive method to forecast future value

public static double forecast(double initialValue, double growthRate, int years) {

// Base case

if (years == 0) {

return initialValue;

}

// Recursive case

return forecast(initialValue, growthRate, years - 1) \* (1 + growthRate);

}

// Iterative method to forecast future value (optimized)

public static double forecastIterative(double initialValue, double growthRate, int years) {

double result = initialValue;

for (int i = 0; i < years; i++) {

result \*= (1 + growthRate);

}

return result;

}

public static void main(String[] args) {

double initialValue = 1000.0; // Starting value

double growthRate = 0.10; // 10% growth per year

int years = 5; // Forecast period

// Using recursive approach

double resultRecursive = forecast(initialValue, growthRate, years);

// Using iterative approach (optimized)

double resultIterative = forecastIterative(initialValue, growthRate, years);

// Print results

System.out.printf("Recursive Result: %.2f\n", resultRecursive);

System.out.printf("Iterative Result: %.2f\n", resultIterative);

}

}

**OUTPUT**

A screenshot of a computer

AI-generated content may be incorrect.