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

Solution:

import java.util.Scanner;

public class Main {

public static double futureValueRecursive(double presentValue, double growthRate, int years) {

if (years == 0) {

return presentValue;

} else {

return (1 + growthRate) \* futureValueRecursive(presentValue, growthRate, years - 1);

}

}

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter Present Value (₹): ");

double presentValue = scanner.nextDouble();

System.out.print("Enter Annual Growth Rate (e.g., 0.08 for 8%): ");

double growthRate = scanner.nextDouble();

System.out.print("Enter Number of Years: ");

int years = scanner.nextInt();

double result = futureValueRecursive(presentValue, growthRate, years);

System.out.printf("Future Value after %d years: ₹%.2f%n", years, result);

System.out.println("\n--- Analysis ---");

System.out.println("Recursive Time Complexity: O(n), where n = number of years.");

System.out.println("Recursion simplifies logic, but may be inefficient for large n due to call stack depth.");

System.out.println("To improve performance, memoization or iteration can be considered (not used here).");

}

}

Output:

A screenshot of a computer program

AI-generated content may be incorrect.