**DEEPSKILLING:**

**Data structure and algorithms:**

**Exercise: Financial Forecasting**

**Step 1: Understand Recursive Algorithms**

Recursion is a programming technique where a method calls itself to solve a problem. It simplifies problems that can be broken down into smaller, similar subproblems.

Example: Calculating future value where each year's value grows by a fixed rate.

**Step 2: Building a Java code**

**public class FinancialForecast {**

**// Recursive method to calculate future value**

**public static double calculateFutureValue(double presentValue, double rate, int years) {**

**if (years == 0) {**

**return presentValue;**

**}**

**return calculateFutureValue(presentValue, rate, years - 1) \* (1 + rate);**

**}**

**public static void main(String[] args) {**

**double presentValue = 10000.0;**

**double rate = 0.08; // 8% annual growth**

**int years = 5;**

**double futureValue = calculateFutureValue(presentValue, rate, years);**

**System.out.printf("Future Value after %d years: %.2f\n", years, futureValue);**

**}**

**}**

**Step 3: Implement Recursive Forecasting Logic**

The method above does this:

* Reduces the problem size each time (years - 1)
* Multiplies the previous year’s value by (1 + rate)
* Stops when years == 0 (base case)

**Step 4: Analysis**

**🔹 Time Complexity**

* The time complexity is **O(n)**, where n = years, since the method is called once for each year.
* The space complexity is also **O(n)** because each recursive call is stored in the call stack.