**DATA STRUCTURES AND ALGORTHMS**

**Exercise 7: Financial Forecasting**

**What is Recursion?**

Recursion is a programming technique where a method calls itself to solve a smaller part of the problem until it reaches a base case. It's useful for problems that can be broken down into similar subproblems.

Example in Financial Forecasting:  
Predict future value by applying a growth rate repeatedly for each year.

FV = PV \* (1 + r)^n

**CODE**

public class Main {

// Recursive method to calculate future value

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

if (years == 0) {

return presentValue;

}

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

}

// Optimized version using iteration

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

double result = presentValue;

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

result \*= (1 + growthRate);

}

return result;

}

public static void main(String[] args) {

double presentValue = 1000; // Initial investment

double growthRate = 0.08; // 8% annual growth

int years = 5; // Forecasting for 5 years

// Recursive calculation

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

System.out.printf("Future Value (Recursive): %.2f\n", futureRecursive);

// Iterative calculation

double futureIterative = futureValueIterative(presentValue, growthRate, years);

System.out.printf("Future Value (Iterative): %.2f\n", futureIterative);

}

}

**Step 4: Time Complexity & Optimization**

**🔹 Time Complexity:**

* Without optimization: **O(n)** — since we make n recursive calls.

**🔹 Optimization:**

* For deeper recursion, use **memoization** or **iterative solution** to prevent **stack overflow** or repeated calculations.
* 