**Algorithms and Data Structures**

**Exercise 7: Financial Forecasting:**

**Main.java:**

import java.util.Scanner;

public class FinancialForecast {

public static double forecastRecursive(int year, double initialAmount, double growthRate) {

if (year == 0) {

return initialAmount;

}

return forecastRecursive(year - 1, initialAmount, growthRate) \* (1 + growthRate);

}

public static double forecastIterative(int year, double initialAmount, double growthRate) {

double result = initialAmount;

for (int i = 1; i <= year; i++) {

result \*= (1 + growthRate);

}

return result;

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.print("Enter initial amount: ");

double initial = sc.nextDouble();

System.out.print("Enter annual growth rate (in %): ");

double rate = sc.nextDouble() / 100;

System.out.print("Enter number of years to forecast: ");

int years = sc.nextInt();

double futureRecursive = forecastRecursive(years, initial, rate);

double futureIterative = forecastIterative(years, initial, rate);

System.out.printf("\n Financial Forecast After %d Years \n", years);

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

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

System.out.println("\nTime Complexity Analysis ");

System.out.println("Recursive: O(n) - Recursively calls for each year.");

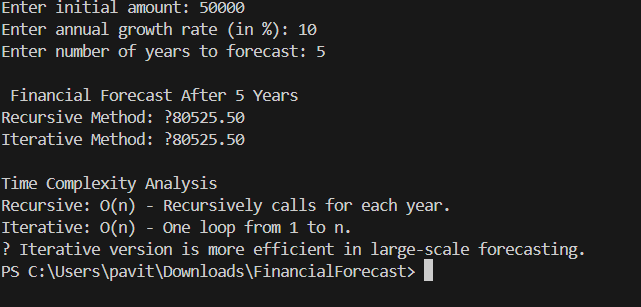
System.out.println("Iterative: O(n) - One loop from 1 to n.");

System.out.println("➡ Iterative version is more efficient in large-scale forecasting.");

}

}

**Output:**

****