**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.**

**Given:**

* An initial value
* A fixed annual growth rate
* A number of years

We want to predict the future value using:

FutureValue(n)=CurrentValue×(1+rate)^n

**CODE SECTION:**  
  
**FinancialForecast.java**  
package DataStruct\_Task7;

public class FinancialForecast {

    public static double forecastRecursive(double currentValue, double growthRate, int years) {

        if (years == 0) {

            return currentValue;

        }

        return forecastRecursive(currentValue \* (1 + growthRate), growthRate, years - 1);

    }

    public static void main(String[] args) {

        double initialValue = 10000;  // ₹10,000

        double growthRate = 0.10;     // 10% per year

        int years = 5;

        double result = forecastRecursive(initialValue, growthRate, years);

        System.out.println("Future Value after " + years + " years: ₹" + result);

    }

}

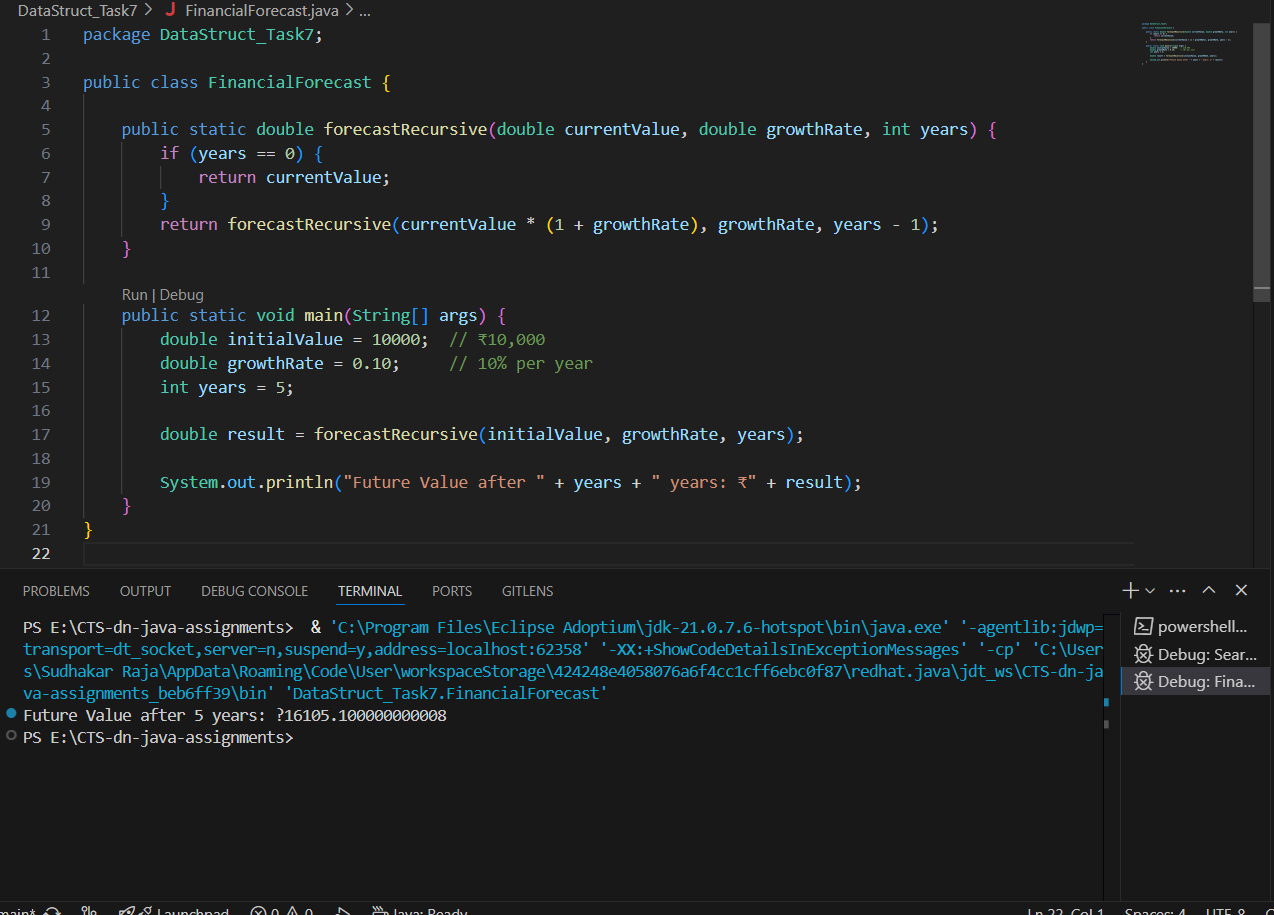
**Analysis**

**🔹 Time Complexity:**

* Each recursive call reduces years by 1 → O(n) time complexity
* Each call performs constant-time computation

**🔹 Space Complexity:**

* Recursive stack → O(n)

**OUTPUT SCREENSHOT:**  
  
  
**Conclusion:**

his hands-on exercise demonstrated the power of **recursion** in simplifying a repeated multiplication-based prediction problem. While the recursive approach is elegant and readable, its linear depth stack calls can be replaced with an iterative loop for better performance in production-grade software.