**Exercise 7: Financial Forecasting**

**Understand Recursive Algorithms**

Recursion is a technique in programming where a method calls itself to solve a problem in smaller sub-parts. It is especially useful in problems that have repetitive patterns or where the current solution depends on previous values. In financial forecasting, recursion can help compute future values based on past growth trends.

**Setup**

We want to calculate the **future value** of an investment based on a **fixed annual growth rate** using recursion.  
Let:

* initialAmount be the investment,
* rate be the growth rate (e.g., 0.08 for 8%),
* n be the number of years.

**Code**

public class FinancialForecasting {

public class FinancialForecasting {

public static double futureValue(double initialAmount, double rate, int years) {

if (years == 0) {

return initialAmount;

}

return futureValue(initialAmount, rate, years - 1) \* (1 + rate);

}

public static void main(String[] args) {

double initialAmount = 10000; // ₹10,000 as the starting investment

double rate = 0.08;

int years = 5;

double result = futureValue(initialAmount, rate, years);

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

}

}

**Analysis**

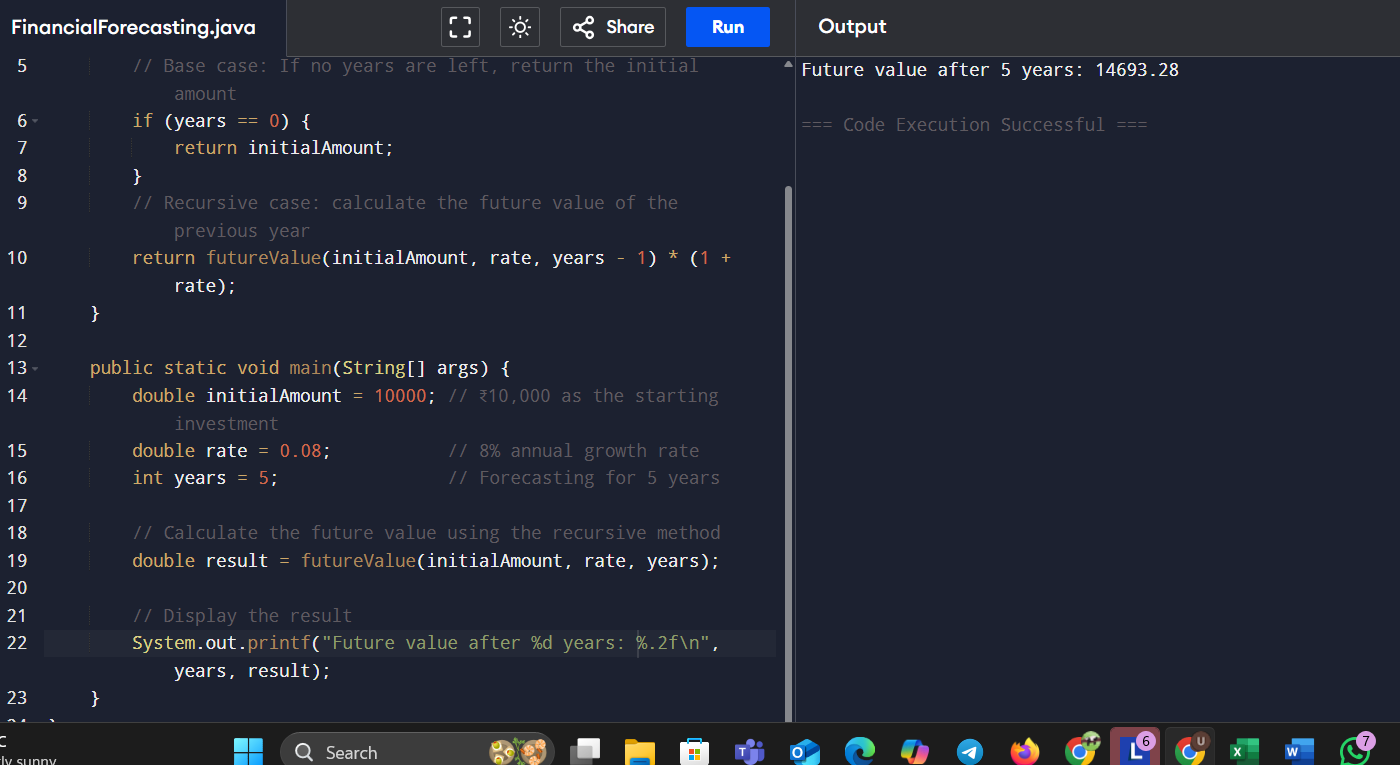
* **Time Complexity:** O(n), where n is the number of years.
* **Space Complexity:** O(n) due to recursive call stack.

**How to Optimize:**

* **Use Iteration:** Avoid recursion to save memory by using a loop instead.
* **Memoization (if needed):** For problems with multiple recursive calls, store already computed values to avoid repetition.

**Conclusion:**

Recursion simplifies the logic for calculating future value, but for large n, an iterative approach is more memory-efficient. This recursive method is a good fit for educational purposes or small-scale forecasting.Top of Form

Bottom of Form