**Financial Forecasting**

### ****Understand Recursive Algorithms****

**Recursion** is when a method calls itself to solve smaller parts of a problem.

Example:  
If you want to calculate future value for year n, you can calculate year n-1 first, then apply the growth rate

**Full Code:**

using System;

namespace FinancialForecastingTool

{

class Program

{

static void Main(string[] args)

{

Console.Write("Enter initial amount: ");

double initialAmount = Convert.ToDouble(Console.ReadLine());

Console.Write("Enter annual growth rate (in %): ");

double growthRate = Convert.ToDouble(Console.ReadLine()) / 100.0;

Console.Write("Enter number of years: ");

int years = Convert.ToInt32(Console.ReadLine());

double futureValue = CalculateFutureValue(initialAmount, growthRate, years);

Console.WriteLine($"\nPredicted value after {years} years: {futureValue:F2}");

}

// Recursive Method

static double CalculateFutureValue(double amount, double rate, int years)

{

if (years == 0)

return amount;

return CalculateFutureValue(amount, rate, years - 1) \* (1 + rate);

}

}

}

### ****Analysis:****

**Time Complexity:**

The time complexity is **O(n)** (linear), where n is the number of years.

Each recursive call reduces years by 1 until 0.

**Optimization:**

Since recursion may lead to **stack overflow** for very large n, we can optimize it using **iteration** or **memoization**.

For this simple financial forecast, iteration is cleaner.