### **Financial Forecasting**

### **Understand Recursive Algorithms**

#### **What is Recursion?**

Recursion is a programming technique where a method calls itself to solve a smaller instance of the original problem. Each recursive call continues to break down the problem until a base case is reached, at which point the solution is built back up.

Recursive algorithms are especially useful when a problem can be divided into smaller sub-problems of the same type. They are often more elegant and easier to understand than iterative solutions for certain problem types, such as tree traversal, mathematical computations (like factorial or Fibonacci series), and forecasting sequences.

### **Analysis**

#### **Time Complexity**

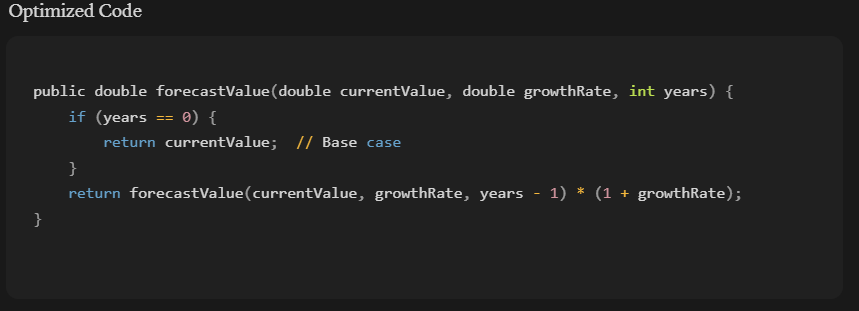
The time complexity of this recursive method is **O(n)**, where n is the number of years being forecasted. This is because each recursive call reduces n by one, and there are no overlapping subproblems or repeated calculations.

In this specific case, recursion does not lead to exponential growth in calls, unlike in algorithms such as the naive Fibonacci implementation.

#### **Optimization Considerations**

While the basic recursive method is efficient enough for small to moderate n, there are a few ways to improve or manage performance:

1. **Memoization:** For problems involving repeated subproblems, memoization can be used to cache and reuse results. However, this isn’t needed for simple linear recursion like the one above.
2. **Tail Recursion:** In languages or environments that support tail call optimization, writing the recursive function in a tail-recursive form can improve performance by reducing stack usage.
3. **Iterative Alternative:** For very large values of n, consider using an iterative solution to avoid stack overflow risk.



In this implementation:

* **Base Case:** When years == 0, the forecast is the current value.
* **Recursive Step:** The function calls itself with one less year, then multiplies the result by (1 + growthRate) to project the next year’s value.

This structure captures the idea that each year’s projection builds on the previous year’s value.

### **Analysis**

#### **Time Complexity**

The time complexity of this recursive method is **O(n)**, where n is the number of years being forecasted. This is because each recursive call reduces n by one, and there are no overlapping subproblems or repeated calculations.