**Week 1 : Data Structures and Algorithms Solution**

**Exercise 7: Financial Forecasting Solution**

Recursion is a programming technique where a method **calls itself** to solve smaller subproblems.

**Key components of recursion:**

* **Base case:** Stops the recursion
* **Recursive case:** Breaks down the problem into smaller parts

**Why recursion?**

It simplifies problems like:

* Tree traversal
* Fibonacci sequence
* Financial growth prediction over time

**Setup: Recursive Future Value Function**

In finance, **Future Value (FV)** with a fixed growth rate is calculated as:

**FV=P×(1+r)^n**

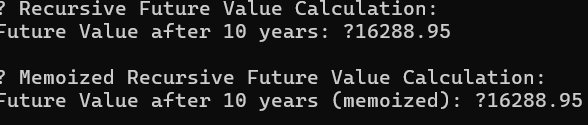
Where:

* P = Present Value
* r = Growth Rate (in decimal, e.g., 0.05 for 5%)
* n = Number of periods (years)

We'll use recursion to compute this by breaking it into:

**FV(n)=FV(n−1)×(1+r)**

**Code is included in the Main.java file   
  
Output :-**

****

**Analysis**

**Time Complexity**

* **Naive recursion:**
  + Time: O(n)
  + Space (call stack): O(n)
* **With memoization:**
  + Time: O(n)
  + Space: O(n) (for memo + stack)

**Optimization Tips**

1. **Memoization (as shown)**: Store previously calculated results to avoid recomputation.
2. **Convert to Iterative**: If recursion depth is large, use a loop to avoid stack overflow.