**Exercise 7: Financial Forecasting**

**Step 1: Understand Recursive Algorithms**

**Recursion** is a method where a function calls itself to solve a problem. Recursive algorithms can simplify complex problems by breaking them down into smaller, more manageable sub-problems. The key components of recursion are:

1. **Base Case**: The condition under which the recursion stops. It prevents infinite recursion and typically provides a direct solution to the simplest form of the problem.
2. **Recursive Case**: The part of the function where it calls itself with modified parameters, aiming to reduce the problem size.

**Advantages of Recursion**:

* Simplifies code and makes it easier to understand for problems that naturally fit a divide-and-conquer approach.
* Can replace complex iterative solutions with more straightforward recursive ones.

**Disadvantages**:

* Can lead to excessive function calls and stack overflow if not managed properly.
* May be less efficient compared to iterative solutions due to overhead from function calls.

**Step 4: Analysis**

**Time Complexity**:

* The time complexity of the recursive algorithm is O(n), where n is the number of periods. Each recursive call reduces the number of periods by 1 until it reaches the base case. This results in a linear number of recursive calls.

**Optimization**: To avoid excessive computation and potential stack overflow, consider the following optimizations:

1. **Memoization**: Store previously computed values to avoid redundant calculations. This technique improves efficiency, especially for problems with overlapping sub-problems.
2. **Iterative Solution**: For simple problems like calculating future values, an iterative solution can be more efficient. Here’s how the iterative approach might look: