**EXERCISE 7**

**FINANCE FORECASTING**

**1. Understanding Recursive Algorithms**

Recursion is a programming technique where a function calls itself in order to solve a problem by breaking it down into smaller, more manageable sub-problems.

Recursion can simplify complex problems by dividing them into smaller, similar problems.

**2. Setup**

To predict future financial values recursively, the problem can be broken down as follows:

* **Base Case**: When there are no more periods left to forecast, the function simply returns the current value.
* **Recursive Case**: For each period, the future value is calculated by applying the growth rate to the present value and then making a recursive call with the updated value and a decremented period count.

The recursive method effectively models the problem of predicting future values as a series of steps, where each step involves computing a new value based on the previous one.

**3. Implementation**

In the implementation, the recursive method repeatedly applies the growth rate to the present value, reducing the number of periods each time it calls itself. The recursion continues until the number of periods reaches zero, at which point the method returns the present value as the future value. This approach is both straightforward and intuitive, allowing the forecast to be computed step by step through recursive function calls.

**4. Analysis**

**Time Complexity**

The time complexity of the recursive algorithm is O(n), meaning it scales linearly with the number of periods (n). Each recursive call does a fixed amount of work, and the total number of calls is directly related to n. Therefore, as the number of periods increases, the time taken by the algorithm increases proportionally.

**Optimization**

Recursive algorithms, while elegant, can sometimes be inefficient due to the overhead of multiple function calls and maintaining the call stack. To optimize:

1. **Iterative Approach**: For problems where the recursion depth can become very large, converting the recursive solution to an iterative one can reduce overhead and improve performance. An iterative approach uses loops to achieve the same result, avoiding the function call stack issue.
2. **Memoization**: Although not strictly necessary for this specific problem, memoization can be used in other recursive scenarios where the same sub-problems are solved multiple times. It involves storing the results of expensive function calls and reusing these results when the same inputs occur again.

In summary, while recursion provides a clear and natural way to solve problems like financial forecasting, optimizing through iterative methods or memoization can be beneficial for performance, particularly with larger datasets or more complex problems.