**Exercise 7: Financial Forecasting**

Recursion is a strategy for addressing problems in which a function calls itself to solve a subset of the original problem. Recursion can be used for situations exhibiting self-similar patterns or dependencies in the context of financial forecasting.

Recursion can simplify complex problems by breaking them down into smaller, more manageable subproblems. However, it's essential to define a base case (termination condition) to prevent infinite loops. Recursion can sometimes be less efficient than iterative approaches due to function call overhead, so it's important to consider performance implications.

Recursive algorithms in financial forecasting have different temporal complexity depending on the implementation and challenge. Performance depends on effective base cases, a finite recursive depth, and avoiding redundant computations. Polynomial, exponential, logarithmic, and linear time are examples of common complexities. It is necessary to examine the particular recurrence relation in order to precisely ascertain the algorithm's temporal complexity. Recursive solutions can be made more efficient by employing strategies like memoization or dynamic programming, which save intermediate results and eliminate needless calculations.

To optimize this, We can go for DP(Dynamic Programming). In which, we store the obtained results in a particular index of the DP array. By this we can avoid the same steps called by the recursive function. For the recursive call which we already found the answer, we just return the answer that is stored in a particular index of the DP array.