**Financial Forecasting**

# **Explain the concept of recursion and how it can simplify certain problems.**

Recursion is a programming technique where a function calls itself to solve smaller instances of the same problem. This continues until it reaches a base case. It is a condition that stops the recursion. This technique allows complex problems to be broken down into simpler, more manageable subproblems.

* **Base Case:** This is the simplest, smallest instance of the problem, which can be solved directly without further recursion. It's essential to prevent infinite recursion and stack overflow.
* **Recursive Case:** This involves breaking the problem into smaller instances and calling the function on these instances. Each recursive call brings the problem closer to the base case.

**Uses of Recursion to Simplify Problems:**

* Recursive algorithms are often used with data structures like trees and graphs.
* Algorithms like quicksort, mergesort, and binary search use recursion to divide a problem into smaller subproblems, solve each subproblem, and then combine the results.
* Problems that involve searching for solutions, like solving mazes, puzzles, or combinatorial problems (e.g., the N-Queens problem), can benefit from recursive backtracking.

# **Discuss the time complexity of your recursive algorithm.**

The complexity of my algorithm is O(n) where ‘n’ is the number of time periods. As the algorithm uses Recursion, so the auxiliary space complexity is O(n) dues to stack usage.

# **Explain how to optimize the recursive solution to avoid excessive computation.**

We can optimize the algorithm by introducing **memoization** which will reduce the redundant computations. We will use HashMap to stored previously computed data and use that for further computation. Thus, we can reduce number of steps significantly. But, this costs extra space of O(n) along with stack memory.