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

**A**. Concept of Recursion:

* Recursion: Recursion is a programming technique where a function calls itself directly or indirectly to solve a problem. It breaks down a complex problem into smaller, more manageable subproblems.
* Base Case and Recursive Case:
  + Base Case: The condition under which the recursion terminates. It prevents the function from calling itself indefinitely.
  + Recursive Case: The part of the function that includes the recursive call. It breaks down the problem into smaller instances and calls the function with these smaller instances.

How Recursion Simplifies Problems:

* Divide and Conquer: Recursion divides the problem into smaller subproblems of the same type. It simplifies the overall problem-solving process by focusing on solving each subproblem.
* Natural Fit for Certain Problems: Problems like tree traversals, factorial calculation, and Fibonacci sequence are naturally suited for recursive solutions because they involve repetitive, self-similar operations.

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

**A.** Time Complexity: The time complexity of the recursive forecast method is O(n), where n is the number of periods (time). This is because the method makes a recursive call for each period, resulting in n calls.

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

**A.** Memoization: One way to optimize the recursive solution and avoid excessive computation is to use memoization. Memoization stores the results of subproblems in a data structure (like an array or a map) so that they can be reused when the same subproblem occurs again.