**Understanding Recursive Algorithms:**  
Recursion is a concept in programming where a function calls itself to solve a problem. The idea is to break down complex problems into smaller, more manageable ones. It often provides a clean and simple way to implement solutions, especially for problems that have repetitive patterns or can be divided into similar sub-problems.

In financial forecasting, where future values depend on past data, recursion can be used effectively. For example, if the value in the next year depends on the current year's value and a fixed growth rate, recursion can be applied to repeatedly calculate values year after year until the target year is reached.

**Analysis**

**Time Complexity:**The time complexity of this recursive algorithm is O(n), where 'n' is the number of years. This is because the function is called once per year, reducing the years by 1 each time.

**Optimization:**  
Though recursion works well here, in some cases with large input sizes, it may cause a stack overflow or lead to repeated computations. To optimize this, memoization (storing results of subproblems) can be used. Alternatively, an iterative version of the same logic can be implemented which avoids recursion completely.