**Understanding Recursive Algorithms**

**Concept of Recursion:**

* Description: Recursion is a programming technique where a function calls itself to solve a smaller instance of the same problem. It often simplifies problems by breaking them down into more manageable subproblems.
* Advantages: Simplifies code for problems that have a natural recursive structure, such as tree traversals, factorial calculations, and the Fibonacci sequence.
* Disadvantages: Can lead to excessive computation and stack overflow if not optimized or if the base case is not defined correctly.

**Analysis**

**Time Complexity**:

* The time complexity of the recursive algorithm is O(n), where n is the number of years. This is because the algorithm makes one recursive call for each year.

**Optimization**:

* **Memoization**: To avoid excessive computation, we can use memoization to store and reuse results of previous calculations. This will improve efficiency, especially if the same subproblems are solved multiple times.
* **Iterative Approach**: Alternatively, converting the recursive algorithm to an iterative one can prevent stack overflow issues and optimize space complexity.