**1. Understand Recursive Algorithms**

**Concept of Recursion**

* **Description:** Recursion is a method of solving problems where a function calls itself as a subroutine. This allows the function to be repeated several times, as it can call itself during its execution.
* **Advantages:**
  + **Simplicity:** Recursion can simplify the code for problems that have a natural recursive structure (e.g., tree traversal, factorial calculation).
  + **Readability:** Recursive solutions are often more elegant and easier to read.
* **Disadvantages:**
  + **Memory Usage:** Recursive functions use more memory due to the call stack.
  + **Performance:** Excessive or deep recursion can lead to performance issues or stack overflow errors.

**4. Analysis**

**Time Complexity of the Recursive Algorithm**

* **Time Complexity:** O(n)
  + **Explanation:** The recursive algorithm makes a single recursive call for each year until years reaches 0. This results in a linear time complexity proportional to the number of years.

**Optimization to Avoid Excessive Computation**

* **Memoization:**
  + **Description:** Memoization is a technique used to store the results of expensive function calls and reuse them when the same inputs occur again. This can significantly improve the performance of recursive algorithms.