Analysis

Recursion:

- Recursion is a method where a function calls itself to solve a problem. It often breaks down a problem into simpler sub-problems of the same type.

- Recursion can simplify the implementation of problems that have a natural hierarchical or repetitive structure. It often leads to cleaner and more intuitive code for problems that can be divided into smaller, similar problems.

Time Complexity:

- **Recursive Time Complexity**: The time complexity of this recursive algorithm is O(n), where `n` is the number of periods. Each recursive call performs a constant amount of work and reduces the problem size by one.

Optimization:

- **Avoid Excessive Computation**: To avoid excessive computation and redundant calculations, memoization (caching previously computed results) can be used. This technique stores results of expensive function calls and reuses them when the same inputs occur again.
- **Tail Recursion**: If the language supports tail call optimization, converting the recursive function to a tail-recursive function can improve efficiency.