1. Understand Recursive Algorithms:

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

Ans: Recursion is a programming technique where a function calls itself to solve a problem. A recursive function typically breaks down a problem into smaller instances of the same problem (Divide and Conquer), eventually reaching a base case that terminates the recursion. Suitable for problems with a recursive nature, such as mathematical calculations, tree traversals, and combinatorial problems.

4. Analysis:

Discuss the time complexity of your recursive algorithm.

The time complexity of the recursive algorithm is O(n), where n is the number of periods. This is because the function makes a recursive call for each period, resulting in n recursive calls in total.

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

Memoization: Stores results of subproblems to avoid redundant computations. Suitable for recursive solutions.

Dynamic Programming**:** Solves problems by combining results of smaller subproblems, either via memoization (top-down) or tabulation (bottom-up).