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

Ans)

Recursion helps in decomposition of complex problems into simpler sub-problems. Each recursive call handles a smaller piece of the problem. In Fibonacci series ,we use recursion by adding 2 numbers before the current number again and again to get the current number.

Recursive solutions often lead to more concise and readable code compared to iterative solutions. Recursion is more preferred in trees, graphs than iterative approach as it is more intuitive.

1. **Time Complexity:**

The time complexity of this recursive algorithm is O(n), where n is the number of periods. This is because each recursive call processes a single period, resulting in a linear number of calls.

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

Ans)

* **Memoization:** Store previously computed results to avoid redundant calculations in problems where overlapping subproblems occur.
* **Tail Recursion:** This specific problem doesn't benefit much from tail recursion optimization, but generally, tail-recursive functions can be optimized by the compiler or interpreter to avoid excessive stack usage.