**Recursion:**

Recursion is a concept in which the problem is divided into multiple sub-problems, and then sub problems are solved first, and the results of sub problems are merged to get the final solution.

Programatically, Recursion is defined as calling a function multiple times inside a function. Such types of functions are also called as recursive functions.

1. Calling of function, here means, solving a sub problem
2. The calling of function multiple times, will end at a base case
3. In each call of the function a logic will be implemented

**Example:**

1. Factorial of a number:
   1. 5! = 5\*4\*3\*2\*1
   2. 4! = 4\*3\*2\*1
   3. General Eqn : factorial(n) = factorial(n-1) \* n
   4. 0! = 1! = 1 (base case)
   5. These types of general equations and base cases are necessary to write a recursive code

Recursion simplifies, certain problems by dividing them into smaller sub-problems and solving them, and merging all the results.

**Time Complexity of Compound Interest Recursion Code in Exercise:**

1. Recursion time complexities, can be calculated using Recurrence-relation
2. Let T(n) be the time for calculating future amount for n years
3. Then, as per our recursion logic
   1. T(n) = 1+T(n-1)
   2. Base case: O(1)
   3. Logic: O(n-1)
4. Solving Recurrence Relation
   1. T(n-1) = 1+T(n-2)
   2. T(n) = 1+(1+(T(n-2))) = 2+T(n-2)
   3. T(n-2) = 1+T(n-3)
   4. T(n) = 2+(1+T(n-3)) = 3+T(n-3)
   5. General Eqn: T(n) = k+T(n-k)
   6. Base case T(0) = 1 (O(1))
   7. Hence, for n=k,
      1. T(n) = n+T(0) => T(n) = n; which is O(n)
5. Hence, time complexity for the recursion forecast code is O(n)
6. Optimization:
   1. There is a simple formula for compound interest
   2. A = P(1+(r/100))^n
      1. A = Amount after n Years
      2. P = Principal Amount
      3. r = growthRate
      4. n = No. of years
7. This would give result in O(1) time.

**General Optimizations for Recursive Solutions:**

1. Memoization: This technique avoids re-calculation of sub-problems which are already computed, hence, optimizing the code.

Since, there are no overlapping sub-problems in out futureForecast function, memorization is not applicable.