**1.Understand Recursive Algorithms**

The process in which a function calls itself directly or indirectly is called recursion and the corresponding function is called a recursive function.

Features:

1. A recursive algorithm takes one step toward solution and then recursively call itself to further move. The algorithm stops once we reach the solution.
2. Since called function may further call itself, this process might continue forever. So it is essential to provide a base case to terminate this recursion process.
3. Makes code **cleaner and more intuitive**, especially for naturally recursive problems (like compounding growth).
4. Removes the need for loops in many scenarios.

Steps of Implementation:

S**tep1 - Define a base case:** Weidentify the simplest (or base) case for which the solution is known or trivial. This is the stopping condition for the recursion, as it prevents the function from infinitely calling itself.  
  
**Step2 - Define a recursive case:** We define the problem in terms of smaller subproblems. We break the problem down into smaller versions of itself, and call the function recursively to solve each subproblem.  
  
**Step3 - Ensure the recursion terminates:** We make sure that the recursive function eventually reaches the base case, and does not enter an infinite loop.  
  
**Step4 - Combine the solutions:** We combine the solutions of the subproblems to solve the original problem.

**4. Analysis**

Time & Space Complexity:

1. Each recursive call reduces years by 1.
2. Total recursive calls= n

Time Complexity: O(n)

Space Complexity: O(n)

Optimization Techniques:

1. Tail Recursion:

public static double calculateFutureValueTail(double pv, double rate, int years, double acc) {

if (years == 0) return acc;

return calculateFutureValueTail(pv, rate, years - 1, acc \* (1 + rate));

}

1. Iterative Approach

public static double calculateFutureValueIterative(double pv, double rate, int years) {

double result = pv;

for (int i = 0; i < years; i++) {

result \*= (1 + rate);

}

return result;

}