**Recursion**

**Concept of Recursion**

Recursion is a technique in programming where a function solves a problem by calling itself with a smaller part of the same problem. It breaks down complex tasks into simpler, manageable units.

**Key Components:**

1. **Base Case**
   * The simplest instance of the problem that stops further recursion.
   * Example: In factorial, 0! = 1 is the base case.
2. **Recursive Case**
   * The part where the function calls itself on a reduced version of the problem.
   * Example: factorial(n) = n \* factorial(n - 1)

**Benefits of Recursion**

1. **Simplifies Complex Problems**  
   Recursion breaks a task into smaller versions, making it easier to handle and implement.
2. **Natural Fit for Some Problems**  
   Certain problems like tree traversal, the Fibonacci sequence, or Tower of Hanoi have recursive structures, making recursion ideal.
3. **Readable Code**  
   Recursive solutions often look cleaner and more logical than their iterative counterparts.

**Limitations**

* **Risk of Stack Overflow**: Deep recursion may exceed call stack limits.
* **Performance**: Repeated calculations can slow performance unless optimized with memoization.
* **Debugging Difficulty**: Tracing errors in recursive calls can be tricky.

**Time and Space Complexity of Recursive Forecasting**

Using memoized recursion to forecast financial values:

* **Time Complexity**: O(n), where *n* is the number of years. Each value is computed once and stored.
* **Space Complexity**: O(n) due to both the call stack and memoization map.

**Optimized Approaches:**

1. **Iterative Method:**

public static double calcFutureValue(double principal, double rate, int yrs) {

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

principal \*= (1 + rate);

}

return principal;

}

Time Complexity: O(n), Space Complexity: O(1)

1. **Mathematical Formula:**public static double calcFutureValue(double principal, double rate, int yrs) {

return principal \* Math.pow(1 + rate, yrs);

}

Time and Space Complexity: O(1)

This formula is the most efficient for compound growth problems when accuracy is sufficient.