Linear recursion is a type of recursion where a function makes at most one recursive call to itself in each execution.

1. **Basic Concept**:
   * Each recursive call creates a new instance of the function
   * The previous instance waits for the result of the new call
   * This forms a linear chain of function calls
2. **Stack Mechanism**:
   * Each function call is pushed onto the call stack
   * The stack grows with each recursive call
   * As base cases are reached, the stack unwinds
   * Results propagate back up through the stack

Ex:

Entering factorial(4)

Need to calculate factorial(3)

Need to calculate factorial(2)

Need to calculate factorial(1)

Base case reached: factorial(1) = 1

factorial(2) = 2 \* factorial(1) = 2

factorial(3) = 3 \* factorial(2) = 6

factorial(4) = 4 \* factorial(3) = 24

1. **Key Points**:
   * Each indentation level represents a new stack frame
   * The base case (n <= 1) stops the recursion
   * Values are calculated during the "unwinding" phase