What is Recursion?

Recursion is a programming technique where a function calls itself to solve a problem by breaking it down into smaller, similar subproblems. It consists of two main parts:

Base Case – A terminating condition that stops the recursion.

Recursive Case – The function calls itself with a modified input, moving toward the base case.

**How Recursion Simplifies Problems:**

Divide and Conquer – Breaks complex problems into smaller, manageable subproblems (e.g., tree traversals, sorting algorithms like QuickSort).

Elegance & Readability – Often results in cleaner, more intuitive code compared to iterative solutions (e.g., Fibonacci sequence, factorial calculation).

Natural Fit for Recursive Structures – Works well with problems involving nested or hierarchical structures (e.g., directory trees, JSON parsing).

## Analysis

### Time Complexity:

Each call reduces years by 1 → runs n times.

So, **Time Complexity = O(n)**

* **Space Complexity = O(n)** (call stack usage)

### Optimization:

### Problem:Recursive calls can cause ****stack overflow**** for large n.