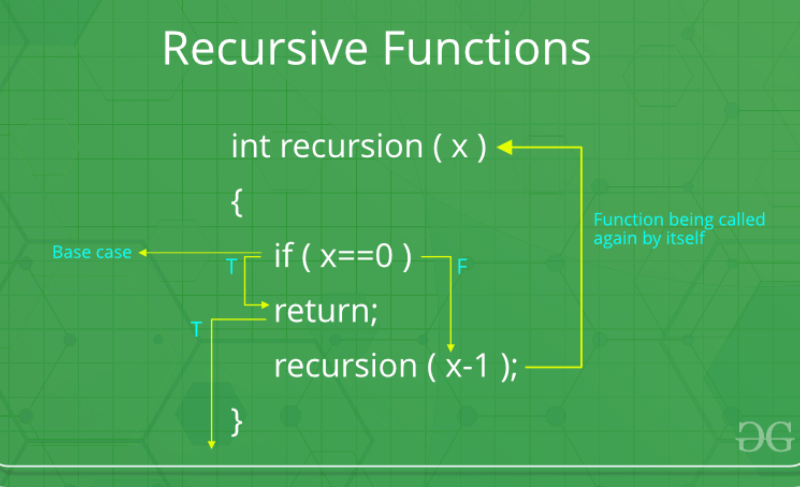
Day 16 of 180 Days — Recursion 🔁

# What is Recursion?

Recursion is when a function calls itself to solve a problem by breaking it into smaller subproblems.

  
Every recursion has:  
1. Base Case – When to stop.  
2. Recursive Case – How to break the problem further.

## Example 1: Reverse a String using Recursion

def reverse\_string(s):  
 if s == "":  
 return ""  
 return reverse\_string(s[1:]) + s[0]

## Example 2: Check if a Number is a Power of 2

def is\_power\_of\_2(n):  
 if n == 1: return True  
 if n <= 0 or n % 2 != 0: return False  
 return is\_power\_of\_2(n // 2)

## Recursion Tree (Example: Power of 2)

8 → 4 → 2 → 1 (Stop)  
If at any point a number is odd (and not 1), recursion stops and returns False.

# Why use Recursion?

- Simplifies complex problems  
- Used in Divide & Conquer algorithms (QuickSort, MergeSort)  
- Essential for Trees, Graphs & Backtracking

***"The journey is long, but the growth is worth it. Let’s keep moving forward! 🔥"***

Created by Veeresh Hanni | Aspiring SDE | 180 Days SDE Preparation Journey | Algorithms365