Recursion

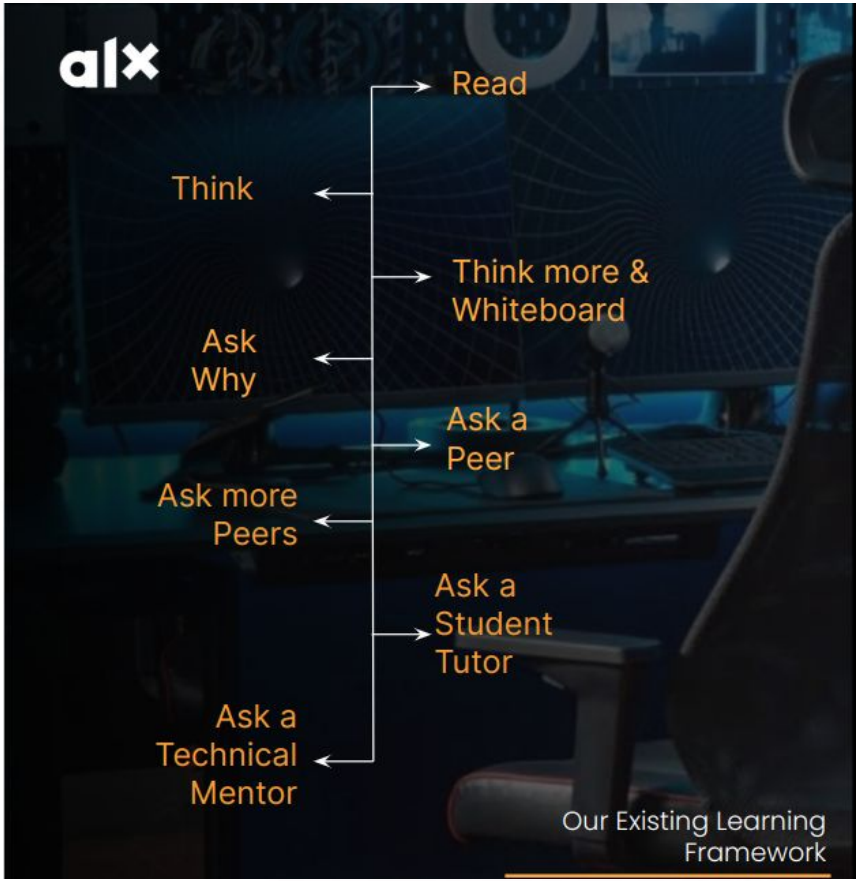
What do you like about Recursion?

How to participate?

Agenda

Item	Time (GMT)
What is Recursion	10
Why use recursion	20
How to write recursive functions (with examples)	15
QA	10

The Framework



What is recursion

- Recursion is a function calling itself.

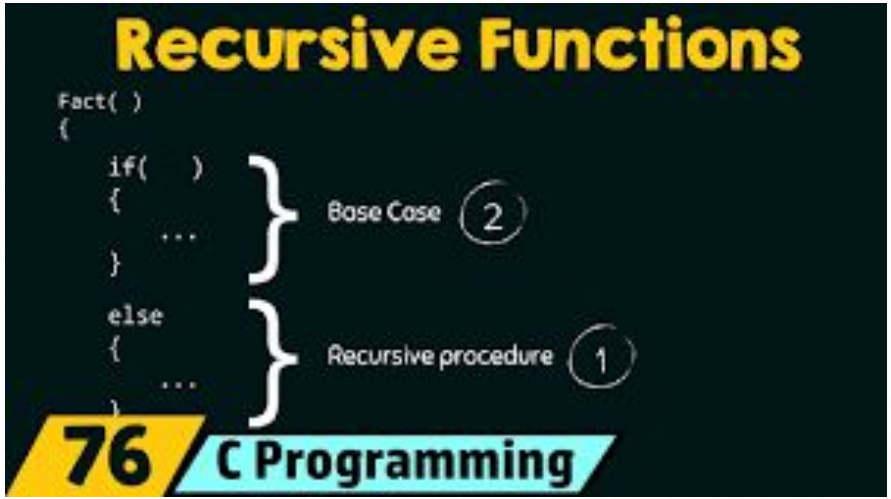
Again What is Recursion

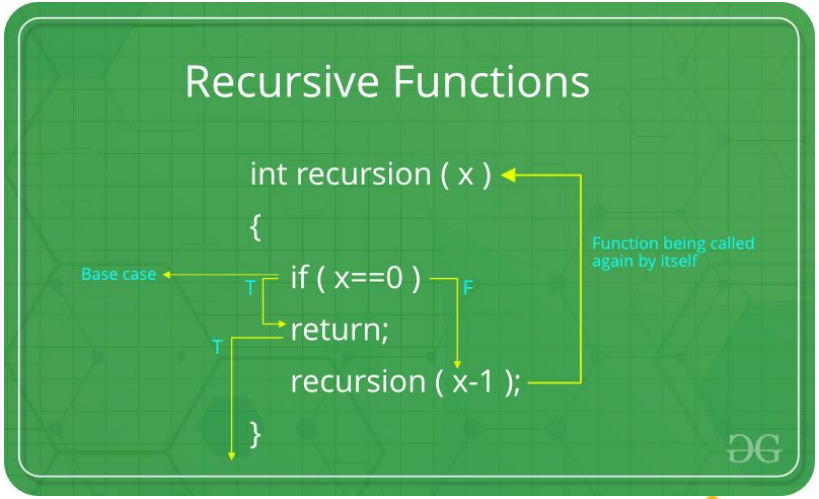
Recursion is the process of defining a problem (or the solution to a problem) in terms of (a simpler version of) itself.

For example, we can define the operation "find your way home" as:

1. If you are at home, stop moving.
2. Take one step toward home.
3. "find your way home".

How do we write recursive functions





Why Recursion Works

In a recursive algorithm, the computer "remembers" every previous state of the problem. This information is "held" by the computer on the "activation stack" or recursion stack.

What is factorial?

fac(0) = 1

fac(1) = 1 * fac(0)

fac(2) = 2 * fac(1) = 2 * 1 = 2

fac(3) = 3 * fac(2) = 3 * 2 = 6

fac(100) = 100 * fac(99)

fac(n) = n * fac(n - 1)

Factorial of a Number

Relation -> F(n) = n * f(n - 1)

Base Case: f(0) = 1

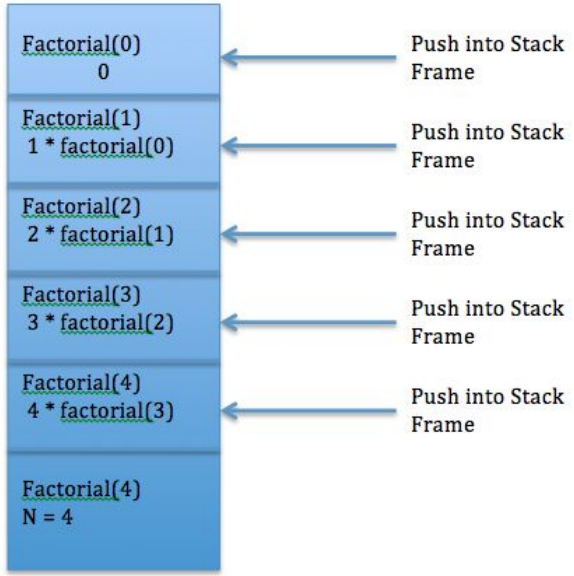
Function factorial(n):

If n == 0:

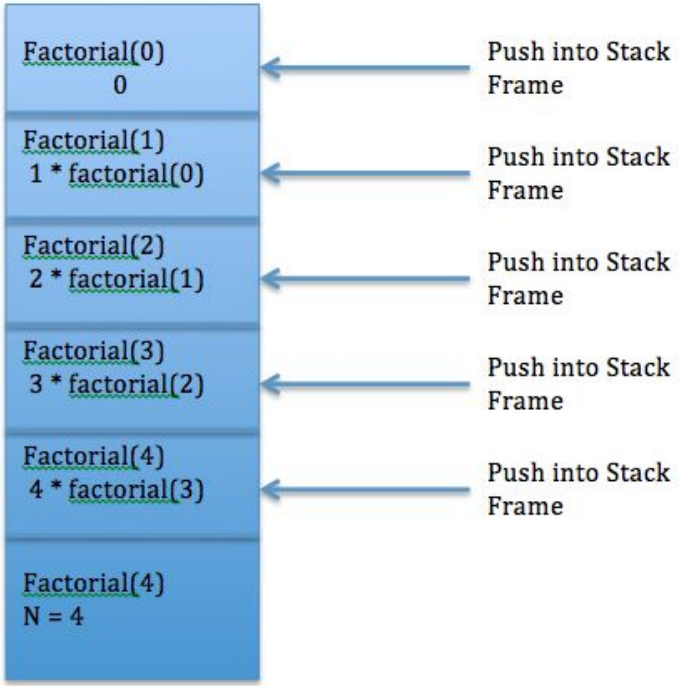
Return 1

Calc = n * factorial(n - 1)

Return calc



Recursion stack



Why do we use Recursion

- Recursion provides a clean and simple way to write code. Some problems are inherently recursive like tree traversals, Fibonacci, etc.

Every recursive code can be written Iteratively

Recursion Visualization

https://recursion.vercel.app/

Fibonacci Sequence

$$f(n) = f(n - 1) + f(n - 2)$$

Disadvantage of Recursion

- Recursive functions are generally slower than non-recursive function.
- It may require a lot of memory space to hold intermediate results on the system stacks.
- Hard to analyze or understand the code.
- It is not more efficient in terms of space and time complexity.

Summary

- There are two types of cases in recursion i.e. recursive case and a base case.
- The base case is used to terminate the recursive function when the case turns out to be true.
- Each recursive call makes a new copy of that method in the stack memory.
- Infinite recursion may lead to running out of stack memory.
- Examples of Recursive algorithms: Merge Sort, Quick Sort, Tower of Hanoi, Fibonacci Series, Factorial Problem, etc.

Resources used

- https://www.geeksforgeeks.org/introduction-to-recursion-data-structure-and-algorithm-tutorials/
- https://www.cs.utah.edu/~germain/PPS/Topics/recursion.html
- https://www.collegenote.net/curriculum/data-structures-and-algorithms/41/454/

Q/A

See you on Next Sessions