**RECURSION**

Geeks For Geeks Explanation: <https://www.geeksforgeeks.org/introduction-to-recursion-data-structure-and-algorithm-tutorials/>

What is Recursion?

Process in which a function calls itself directly or indirectly, the corresponding function is called a recursive function.

Example Problems:

* [Towers of Hanoi (TOH)](https://www.geeksforgeeks.org/c-program-for-tower-of-hanoi/)
* [Inorder/Preorder/Postorder Tree Traversals](https://www.geeksforgeeks.org/tree-traversals-inorder-preorder-and-postorder/)
* [DFS of Graph](https://www.geeksforgeeks.org/depth-first-search-or-dfs-for-a-graph/)

A recursive function solves a particular problem by calling a copy of itself and solving smaller subproblems of the original problems.

**A certain case should be provided in order to terminate a recursion process!**

Need of Recursion

Recursion is one of the best solutions for a task that can be defined with a similar subtask (i.e. the Factorial of a number)

Recursion Properties

* Performing the same operations multiple times with different inputs
* In every step, we try smaller inputs to make a smaller problem
* A **base condition** is needed to stop the recursion

Mathematical Interpretation

Recursive adding – Instead of f(n) = 1 + 2 + 3 + … + n

f(n) = 1 n= 1

f(n) = n + f(n-1) n>1

* f() is being called inside of itself so it is recursive

Recursive Functions in Memory

More memory is used because a recursive function adds to the stack with each recursive call and keeps the values there until the call is finished.

* Uses LIFO structure like the stack data structure

Base Condition

The solution to the base case should be provided and the solution to the bigger problem should be expressed in terms of smaller problems

Example:

int fact(int n) {

if (n <=1) //base case

return 1;

else

return n\*fact(n-1);

}

The base case is n <= 1, and the solution is 1. The larger value of a number can be solved by converting to a smaller one until the base case is reached.

Stack Overflow

If the base case is undefined or never reached, stack overflow may occur.

* wrong base case can cause this

Direct vs Indirect Recursion

Direct Recursive: Function that calls itself

Indirect Recursive: Function that calls a different function that calls the original function (Function A calls Function B and vv)

Tailed vs Non-Tailed Recursion

Tail Recursive: Recursive call is the last to be executed by the function

Memory Allocation to Different Function Calls

A function called from main(), the memory is allocated to it on the stack.

Recursion vs Iteration

|  |  |
| --- | --- |
| Recursion | Iteration |
| Terminates when the base case becomes true | Terminates when the condition becomes false |
| Used with functions | Used with loops |
| Every recursive call needs extra space in the stack memory | Every iteration does not require any extra space |
| Smaller code size | Larger code size |

Summary

* Two case types: recursive case & base case
* Base case is used to determine when the recursive function terminates (when base case is true)
* Each recursive call makes a copy of that method in the stack memory
* Infinite recursion may lead to running out of stack memory
* Examples
  + Merge Sort
  + Quick Sort
  + Towers of Hanoi
  + Fibonacci Series
  + Factorial Problem

Khan Academy Explanation: <https://www.khanacademy.org/computing/computer-science/algorithms/recursive-algorithms/a/recursion>

Recursion is like Russian Nesting Dolls

* The large doll has many smaller dolls within it, until it reaches one so small that it can not contain another

Factorial Function

The factorial of n (n!) is the product of the integers 1 through n. (5! = 5 \* 4 \* 3 \* 2 \* 1 = 120)

* useful for trying to count how many different orders there are for things or how many different ways there are to combine things
* useful to count how many ways you can choose things from a collection of things

Factorial function is defined for all positive integers, along with 0. 0! = 1

For n! if n is positive it is 1 \* 2 … (n-1) \* n and if n=0 it is 1