1. Function Definition in C:

A function in C consists of:

- Return Type: Specifies the type of value the function returns. Use void if it doesn't return anything.
- Function Name: A unique identifier for the function.
- Parameter List: Types and names of parameters (inputs).
- Function Body: Contains the code to be executed.

Function Declaration (Prototype):

- Declaring a function before using it ensures the compiler knows its signature:

Function Call:

- You call a function by writing its name and passing required arguments:

Return Statement:

- The `return` keyword exits a function and optionally returns a value.

Performing Arithmetic Operations with Numeric Variables in C:

- Division with Integers vs. Floats:
  Integer division discards the decimal part. Use floating-point variables for accurate division results.
- Modulus Operator %:
  Only works with integers, not floats.
- Operator Precedence:
  Multiplication, division, and modulus have higher precedence than addition and subtraction. Use parentheses to enforce order.

What Recursion is :

Recursion in C (and in programming generally) is a technique where a function calls itself to solve a problem. It breaks down complex problems into simpler sub-problems until a base condition is met, preventing infinite loops.

A recursive function must have two key components:

- Base Case (Termination Condition): Stops the recursion when a condition is met.
- Recursive Case: Function calls itself with a modified argument, moving closer to the base case

Types of Recursion:

Direct Recursion vs. Indirect Recursion

Direct Recursion:

A function directly calls itself to solve a problem.

Indirect Recursion:

A function calls another function, which then calls the original function.

Tail Recursion vs. Head Recursion:

Tail Recursion:

The recursive call is the last operation performed before returning. It's easier for compilers to optimize (tail-call optimization).

Head Recursion:

The recursive call occurs before any operations. Execution happens in reverse order after returning from recursion.

Linear Recursion vs. Tree Recursion:

Linear Recursion:

Each function call leads to a single recursive call, forming a linear chain.

Tree Recursion:

A function calls itself multiple times, leading to a tree of function calls. This can cause exponential growth in calls.

Nested Recursion:

A recursive function's argument is the result of another recursive call. This is rare and complex.

Mutual Recursion:

Two or more functions call each other in a cyclic manner. It's a form of indirect recursion.

Problems on code force:

Wonderful number:

https://codeforces.com/group/MWSDmqGsZm/contest/223205/submission/312700584

print 1 to N:

https://codeforces.com/group/MWSDmqGsZm/contest/223339/submission/312818331

Print Digits using Recursion :
https://codeforces.com/group/MWSDmqGsZm/contest/223339/submission/312821598

Fibonacci:

https://codeforces.com/group/MWSDmqGsZm/contest/223339/submission/312824658

3n + 1 sequence:

https://codeforces.com/group/MWSDmqGsZm/contest/223339/submission/312845032