**FUNCTIONS IN C**

A function is a block of code that performs a specific task, which can be reused multiple times in a program. Functions allow for better code organization, modularity, and reusability.

**SYNTAX OF C**

return type function name(parameters)

{

// code to be executed

return value; // optional

}

**Key Components of a Function in C:**

***Return Type:*** Specifies the type of value the function will return. Common return types include:

*int (integer)*

*void (no return value)*

*float, double, char, etc.*

***Function Name:*** A unique identifier for the function, used to call it.

***Parameters:*** Variables passed to the function. These parameters receive the values (arguments) when the function is called.

**Function Body:** Contains the code that performs the task. It is enclosed in curly braces {}.

**Return Statement:** The return keyword sends a value back to the calling function. If the function is of type void, no return statement is necessary.

**Types of Functions:**

**Library Functions:** Predefined functions in C libraries, like printf(), scanf(), sqrt(), etc.

**User-Defined Functions:** Custom functions created by the programmer.

**Function Declaration vs. Definition:**

***Function Declaration:*** Informs the compiler about the function name, return type, and parameters. It usually appears before the main function.

*int add(int a, int b); // Declaration*

**Function Definition:** Provides the actual body of the function where the logic resides.

**int add(int a, int b) { // Definition**

**return a + b;**

**}**

**Example of Void Function:**

#include <stdio.h>

void printMessage() {

printf("Hello, World!\n");

}

int main() {

printMessage(); // Function call

return 0;

}

This function doesn't return any value since its return type is void.

**Parameter passing methods**

* Pass by Value
* Pass by Reference (using pointers)

**Advantages of using functions in C**

* Modularity - Functions help divide a complex program into smaller, more manageable sub-programs or modules.
* Code Reusability - Functions allow you to write a block of code once and use it multiple times by simply calling the function.
* Improves Readability - Code written in functions is more organized and easier to follow. Each function represents a logical block, making it clearer what each part of the code does
* Easier Debugging and Maintenance - If there is an issue in the program, debugging is easier when the code is structured into functions.
* Reduces Program Size - By using functions, redundant code is minimized, which reduces the overall size of the program.