# **📘 Functions in C++**

👉 A **function** is a block of code that performs a specific task.

* It helps in **reusability** (write once, use many times).
* It makes the program more **organized** and **readable**.

## **🔹 1. Types of Functions**

### **(a) Library Functions**

* Already defined in C++ libraries.

Example:  
  
 #include <cmath>

cout << sqrt(25); // 5

cout << pow(2, 3); // 8

### **(b) User-defined Functions**

* Created by the programmer.

Example:  
  
 int add(int a, int b) {

return a + b;

}

## **🔹 2. Structure of a Function**

A function in C++ has **3 parts**:

1. **Function Declaration (Prototype)**
   * Tells the compiler about the function.

int add(int, int); // declaration

1. **Function Definition**
   * Actual code of the function.

int add(int x, int y) {

return x + y;

}

1. **Function Call**
   * Executes the function.

int main() {

cout << add(5, 3); // calling the function

}

## **🔹 3. Function Syntax**

returnType functionName(parameters) {

// body of function

return value; // optional

}

Example:

int square(int n) {

return n \* n;

}

## **🔹 4. Types Based on Arguments & Return**

**No arguments, no return** void greet() {

cout << "Hello!";

}

**With arguments, no return** void display(int n) {

cout << "Number: " << n;

}

**No arguments, with return** int getNumber() {

return 10;

}

**With arguments, with return** int add(int a, int b) {

return a + b;

}

## **🔹 5. Recursive Function**

A function that **calls itself**.  
 Example: Factorial

int factorial(int n) {

if(n == 0) return 1;

return n \* factorial(n-1);

}

## **🔹 6. Inline Function**

* Suggests compiler to replace function call with function code → makes execution faster for small functions.

inline int square(int x) {

return x \* x;

}

# **✅ Example Program**

#include <iostream>

using namespace std;

// function declaration

int add(int, int);

// function definition

int add(int a, int b) {

return a + b;

}

int main() {

int x, y;

cout << "Enter two numbers: ";

cin >> x >> y;

cout << "Sum = " << add(x, y); // function call

return 0;

}

# **📌 Summary**

# **Functions** make code reusable, structured, and easier to debug.

# **Library functions** → already built-in (sqrt, pow).

# **User-defined functions** → written by programmer.

# **Types** → Based on arguments & return values. Special ones → **Recursive** and **Inline** functions.