Function

Opentechz Pvt Ltd .
By Parthasarathi Swain

What is a Function?

- A block of code that performs a specific task
- Executes only when called
- Can take inputs (parameters)
- Can return an output (return value)
- Helps to avoid code repetition
- Makes code modular and readable
- Supports top-down programming

Example :

```
int add (int a, int b) {
   return a + b;
}
```

Why Use Functions?

- Avoids writing the same code again and again
- Breaks the big program into smaller parts
- Makes code easy to read and debug
- Promotes modular programming

Types of Functions

Library Functions

- Built-in, provided by C++
- Ready to use – just include the correct header file

Example:

sqrt() – Finds square root

strlen() – Finds string length

pow(), abs(), toupper()

User-defined Functions

- Created by the programmer
- Used to perform specific/custom tasks
- Increases code reusability and clarity

Example:

```
int add(int a, int b) {
        return a + b;
}
```

Library Functions

Function	Purpose	Header File
sqrt(x)	Finds square root of x	<cmath>
pow(x, y)	Raises x to power y	<cmath>
abs(x)	Returns absolute value of x	<cstdlib>
ceil(x)	Rounds x up to nearest integer	<cmath>
floor(x)	Rounds x down to nearest integer	<cmath>
max(a, b)	Returns maximum of a and b	<algorithm>
min(a, b)	Returns minimum of a and b	<algorithm>
strlen(s)	Returns length of string s	<cstring>
strcpy(a, b)	Copies string b to a	<cstring>
strcmp(a, b)	Compares two strings	<cstring>
toupper(c)	Converts character to uppercase	<cctype>
tolower(c)	Converts character to lowercase	<cctype>
rand()	Generates a random number	<cstdlib>
srand(seed)	Sets seed for random number generator	<cstdlib>

User-defined Functions

Parts of a Function
Declaration – Tells compiler about the function
Definition – Actual code block
Calling – When function is used in main()

Function Syntax

```
return_type    function_name (parameter_list) {
    // code to execute
}
```

```
int add(int a, int b) {
    return a + b;
}
```

Function Declaration

- Tells compiler about function name, return type, and parameters
  Example:

```
int add(int, int);
```

Function Definition

- Has the full code of the function
  Example:

```
int add(int a, int b) {
        return a + b;
}
```

Function Call

- Used to **run/execute** a function
- Function must be **called from main()** or another function
- Call happens using the **function name and arguments (if any)**
- Order of execution depends on the **call, not position in code**
- Function can be called **multiple times**

Function Call Example

```
#include <iostream>
using namespace std;

// Function to add two numbers
int add(int a, int b) {
    return a + b;
}
// Function to print a welcome message
void welcome() {
    cout << "Welcome to Function Demo!\n";
}
```

```
int main() {

   // Function call
    welcome();

   // Function call with return
    int sum = add(10, 20);

    cout << "Sum = " << sum << endl;

    return 0;
}
```

Call by Value vs Call by Reference

Call by Value

- A **copy** of the variable is passed to the function
- **Original value remains unchanged**
- **Safe**, but memory-inefficient for large data

Example :

```
void update(int age) {
   age = 30;
}
```

```
int main() {
   int a = 25;
   update(a);      // a is still 25
   cout << a;      // Output: 25
}
```

Call by Value vs Call by Reference

Call by Reference

- A **reference (address)** of the variable is passed
- **Function can modify** the original value
- Useful when changes must reflect back

Example :

```
void update(int &age) {
    age = 30;
}
```

```
int main() {
    int a = 25;
    update(a);       // Now a becomes 30
    cout << a;       // Output: 30
}
```

Recursion (Advanced Concept)

- A function that **calls itself**
- Used to **solve complex problems** by breaking them into smaller sub-problems
- Must have a **base condition** to stop recursion
- If base condition is missing → **infinite loop / stack overflow**

```
int factorial(int n) {
        if (n == 0) {
            return 1;
        }      // Base case

   return n * factorial(n - 1);   // Recursive call
}
```

Tips to Remember

- Always declare functions before main()
- Use **meaningful names**
- Keep functions short and to the point
- Test each function separately

Practice Time

1. Write a function to find square of a number
2. Create a function to print even numbers
3. Create a function to find largest of 3 numbers
4. Recursive function for factorial

OTZ

OPENTECHZ

A dream to realize your dream

Thank You