

Chapter 10: Functions

The process of breaking large programs into smaller sub programs is called **modularisation**. The sub programs are generally called functions.

Merits of modular programming




- Reduces the size of the program.
- Less chance of error occurrence.
- Reduces programming complexity.
- Improves reusability.

Function is a named unit of statements in a program to perform a specific task as part of the solution.

Predefined functions

C++ provides a number of functions for various tasks. While using these functions, some of them require data for performing the task assigned to it. We call them **parameters** or **arguments** and are provided within the pair of parentheses of the function name. There are certain functions which give results after performing the task. This result is known as **value returned** by the function. Some functions do not return any value; rather they perform the specified task.

Type	Functions	Syntax / Example	Operation
String Functions (string.h)	strlen()	strlen(string)	To find the length of a string.
	strcpy()	strcpy(string1, string2)	To copy one string into another
	strcmp()	strcmp(string1, string2)	To compare two strings. <ul style="list-style-type: none"> • Returns 0 if string1 and string2 are same. • Returns a -ve value if string1 is alphabetically lower than string2. • Returns a +ve value if string1 is alphabetically higher than string2.
Mathematical Functions (math.h)	abs()	abs(int)	To find the absolute value of an integer.
	sqrt()	sqrt(double)	To find the square root of a number.
	pow()	pow(double, int)	To find the power of a number. It takes two arguments x and y . Returns the value of x^y .

Character Functions (ctype.h)	isupper()	isupper(char)	To check whether a character is in upper case or not. The function returns non-zero if the given character is in uppercase, and 0 otherwise.
	islower()	islower(char)	To check whether a character is in lower case or not. The function returns non-zero if the given character is in lowercase, and 0 otherwise.
	isalpha()	isalpha(char)	To check whether a character is alphabet or not. The function returns non-zero if the given character is an alphabet, and 0 otherwise.
	isdigit()	isdigit(char) 	To check whether a character is digit or not. The function returns non-zero if the given character is a digit, and 0 otherwise.
	isalnum()	isalnum(char)	To check whether a character is alphanumeric or not. The function returns non-zero if the given character is alphanumeric, and 0 otherwise.
	toupper()	toupper(char c)	This function is used to convert the given character into its uppercase.
	tolower()	tolower(char c)	This function is used to convert the given character into its lowercase.
Conversion Function (stdlib.h)	itoa()	itoa(int n, char c[], int len)	This function is used to convert an integer value to string type. It requires three arguments. The first one is the number to be converted. The second argument is the character array where the converted string value is to be stored and the last argument is the size of the character array.
	atoi()	int atoi(char c[]);	This function is used to convert a string value to integer. It takes a string as argument returns the integer value of the string.
I/O Manipulating Function (iomanip.h)	setw()	char s[]="hello"; cout<<setw(10)<<s <<setw(10)<<"friends"; The output of the given code will be: hello friends	This function is used to set the width for the subsequent string. The word hello will be displayed right aligned within a span of 10 character positions. Similarly, the word friends will also be displayed right aligned within a span of 10 character positions.

User defined functions

The syntax of a function definition is given below:

```
data_type function_name(argument_list)
{
    statements in the body;
}
```

The `data_type` is any valid data type of C++. The `function_name` is a user-defined word (identifier). The `argument_list`, which is optional, is a list of parameters, i.e. a list of variables preceded by data types and separated by commas. The `body` comprises of C++ statements required to perform the task assigned to the function.

A **function prototype** is the declaration of a function by which compiler is provided with the information about the function such as the name of the function, its return type, the number and type of arguments, and its accessibility. The following is the format:

```
data_type function_name(argument_list);
```

Arguments or parameters are the means to pass values from the calling function to the called function. The variables used in the function definition as arguments are known as **formal arguments**. The constants, variables or expressions used in the function call are known as **actual (original) arguments**. If variables are used in function prototype, they are known as dummy arguments.



Two type of Function Calling


Call by Value Method	Call by Reference Method
<ul style="list-style-type: none">• Ordinary variables are used as formal parameters.• Actual parameters may be constants, variables or expressions.• The changes made in the formal arguments do not reflect in actual arguments.• Exclusive memory allocation is required for the formal arguments.	<ul style="list-style-type: none">• Reference variables are used as formal parameters.• Actual parameters will be variables only.• The changes made in the formal arguments do reflect in actual arguments.• Memory of actual arguments is shared by formal arguments.

Recursion is the process of calling a function by itself and the function is known as **recursive function**.

Questions from Previous Years' Question Papers

1. Which one of the following is NOT equal to others?
a) pow(64, 0.5) b) pow(2,3) c) sqrt(64) d) pow(3,2) (1) (July 2017)
2. Write a recursive C++ function that returns sum of the first n natural numbers.
(2) (July 2017)
3. List any three string functions in C++ and specify the value returned by them.
(3) (July 2017)
4. Name the built in function to check whether a character is alphanumeric or not.
(1) (March 2017)
5. Read the function definition given below. Predict the output, if the function is called as convert(7);

```
void convert(int n)
{
    if (n>1)
        convert(n/2);
    cout<<n%2;
}
```


(2) (March 2017)
6. Explain the difference between call-by-value method and call-by-reference method with the help of examples.
(3) (March 2017)
7. Arguments used in call statement are formal arguments. State true or false.
(1) (Sept. 2016)
8. Differentiate the string functions strcmp() and strcmpi().
(2) (Sept. 2016)
9. Differentiate break and continue statements in C++.
(2) (Sept. 2016)
10. Explain recursive function with the help of a suitable example.
(3) (Sept. 2016)
11. Explain Call by Value and Call by Reference methods of function calling with the help of suitable examples.
(4) (Sept. 2016)
12. Suggest most suitable built-in function in C++ to perform the following tasks:
(a) To find the answer for 5^3 .
(b) To find the number of characters in the string "KERALA".
(c) To get back the number 10 if the argument is 100.
(2) (March 2016)
13. A function can call itself for many times and return a result.
(a) What is the name given to such a function? (1)
(b) Write a function definition of the above type to find the sum of natural numbers from 1 to N . (Hint: If the value of N is 5, the answer will be $1+2+3+4+5=15$)
(3) (March 2016)
14. Explain two types of variable according to its scope and life.
(2) (Sept. 2015)
15. Write a C++ Program to display the simple interest using function. (3) (Sept. 2015)

Computer Science - XI

16. The function which calls itself is called a _____. (1) (March 2015)
17. Construct the function prototype for the following functions:
- (a) The function Display() accepts one argument of type *double* and does not return any value.
 - (b) Total() accepts two arguments of type *int*, *float* respectively and return a *float* type value. (2) (March 2015)
18. Name the different methods used for passing arguments to a function. Write the difference between them with examples. (3) (March 2015)

