Programming Language

1. C language

This section introduces you to the C programming language, its origin, history, and significance. Additionally, you can explore the features, applications, execution, general structure and keywords of C.

- Introduction to C
- History of C
- Features of C
- Application areas of C
- Execution flow of C program
- Other translators
- Structure of C program with example
- Keywords

2. Basic concepts

C language has certain elements that you must know before you start coding. This section covers the basics of C programming, including data types (Char, int, void), variables, constants, identifiers, input and output functions, and more.

- Tokens
- Identifiers
- Constants
- Variables
- Data types
- Input and output functions
- Qualifiers
- Modifiers
- Escape sequences

3. Operators and Expressions

Operators and expressions instruct the machine to perform an arithmetic or logical operation on data and variables. To understand the function of each symbol, you need to thoroughly go through this section that covers all the operators and expressions used in a C program.

- Arithmetic operators
- Relational operators
- Logical operators
- Assignment operators
- Increment & Decrement operators
- Conditional/ternary operator
- Bitwise operator
- Size of operator
- Comma operator
- Operators' Precedence and associativity

- Expressions
- Evaluation of expressions

4. Control structures

C program allows you to use structures that can print statements iteratively until it meet a specified instruction. This section will cover the three important loop structures and the essential statements (Goto, Break and Continue) that can control the execution of instructions automatically.

- While
- For
- Do.. While
- Goto Statement
- Break and Continue Statement

5. Control/Decision-Making Statements

A program can use statements that allow the machine to decide when to execute a statement as per instruction. This requires you to understand different decision-making statements where if a condition is true, execution of an immediate instruction happens otherwise, another statement gets executed. You will be introduced to such statements in this section of the C programming course curriculum.

- Simple if
- if..else
- Nested if
- if..else ladder
- Switch..Case statement

6. Math.h Library

The math.h library holds different mathematical functions that, if you use them within a program, directly instruct the machine on what mathematical operation to perform. For example, sqrt() is used for square root operation, log() for natural logarithm of a number. This part of the course will help with advanced c language programming.

- abs(int x)
- floor()
- ceil()
- sqrt()
- pow()
- exp()
- log() and etc.…

7. Arrays

Arrays can store more than one variable of the same data type, but you must understand their syntax properly. This section will give an overview of types of arrays and how they are used in C programming.

- Introduction to arrays
- Types of arrays
- 1d array
- 2d array (matrix)

8. string.h library

As a C programmer, you need to know different string functions and the string.h library contains a set of functions that perform various string operations. This course will introduce you to some standard functions in the library and their use in a C program.

- strlen(str)
- strcpy(des_str, src_str)
- strcat(desc_str, src_str)
- strrev(str)
- strcmp(str1,str2)
- strlwr()
- strupr()

9. Functions

You don't need to repeat the same instruction for the same tasks within a C program, instead use functions. This section will introduce you to different functions and their purposes.

- Function types
- Built-in functions
- User-defined functions
- Recursive functions
- call by value and call by reference

10. Recursions

Recursion is the process where a function calls itself either directly or indirectly. This section of the course will not only clear your basic knowledge but also implement the concept of recursion into real C programs.

- Find the factorial of a given number by using a recursive function
- Find the sum of first n natural numbers by using recursion
- Find the gcd of two numbers by using recursion
- Find the root digit of the number by using recursion
- Program to calculate the power using recursion

11. Storage classes

You must understand where all your variables are getting stored and how long you can access them. This section will teach you the four main storage classes in C and define how they are different from each other.

- Auto
- Static
- Extern
- Register

12. User defined datatypes

Users define some data types and are mostly derived from existing data types; these are user-defined data types. They enable customisation in your C code and implement functions like encapsulation. You will learn the four types of user-defined datatypes in this section of the curriculum.

- Structure

- Union
- enum
- typedef

13. Pointers

Pointers are variables that hold the address of a location that contains another variable. This section gives a brief overview of pointers, their types, syntax and implementation in C programming.

- Pointer types
- Void pointer
- Null pointer
- Wild pointer
- Dangling pointer
- Array of pointers
- Pointer to pointer
- Pointer arithmetic

14. Dynamic memory allocation

According to the rules of C programming, you need to specify the number of elements in an array at compile time. Dynamic memory allocation allows programs to get more memory space while compiling or releasing it if not used. This section will guide you through the functions used for adding, deleting or rearranging data during the runtime.

- malloc()
- calloc()
- realloc()
- free()

15. Files

Once you terminate a c program, your entire data gets lost to preserve that data, you need to store it in a file. This section will give you an in-depth view of files used in C and its relative functions.

- Concept of a file
- Streams
- Text File and Binary Files
- Opening and closing files
- File input/output functions
- Formatted input-output functions
- Character input-output functions

16. Others

Here are a few things that you must know, as without them, your C compiler will keep showing errors when you execute a program.

- Command Line Arguments
- const
- preprocessor directive statements

C programming course curriculum at a glance

C programming course curriculum comprises all core topics covering C's syntax, data types, program structure, pointers, functions, operators, loops, libraries, arrays, and strings. If you are starting new with programming or entering into BCA, MCA, BE, and BTECH courses, C programming is a must-to-learn subject. It makes your programming base clear and prepares you for higher-level coding languages.

C language course subjects and topics to learn

1. Basics of C Program

You must first understand the need for C programming and its history and evaluation. Secondly, you need to focus on the structure of a C program. There are multiple sections in a C program, including documentation, header files, definition, global variables declaration, main program, and user-defined function sections. All of these have different purposes and functions within a C program. You must understand them before starting to code.

But understanding the structure is not sufficient, you need to understand a few more things.

- Variables and their declaration syntax
- Datatypes of C (Char, int, float, double, void)

2. Operators in C language

Operators used in C programs tell the machine to perform mathematical or logical manipulation of data and variables. There are multiple operators used in writing a C program, and each has its function. These are :

- Arithmetic operators(+,-,*,/,%)
- Relational operators(<,<=,>,>=,==,!=)
- Logical operators(&&, !!, !)
- Bitwise operators
- Assignment operators (=)
- Ternary/Conditional operator
- Increment/Decrement operators (++a, a++, –a, a–)
- Comma operator
- Size of operator

3. Looping Statements in C

Looping statements in C makes coding easy and faster. When you put something inside a loop, the instructions keep executing until a specific condition is reached. Such statements control how many times a particular operation or sequence of operations needs to be performed in succession. These loops are classified as

- entry-control loop (for, while)
- exit-control loop (do-while)

4. Functions

A function in C programming is a self-contained block of statements that instructs the machine to perform coherent tasks of some kind. They can be built-in functions or user-defined functions. You need to understand these functions, especially when you're doing C projects where you need to perform the same calculation or similar tasks multiple times. Instead of writing the same line of code multiple times, using functions can be helpful. Here, you need to cover a few things, like

- Function types
- Components of functions
- Call by value and call by reference

5. Recursion

In C programming, recursion is the technique of how a function can call itself. This can be classified as direct and indirect recursions, where either a function calls itself directly or uses another function to call itself. You need to learn these techniques separately so that you do not implement the wrong base condition for a recursive call and confine it into an infinite loop. Here, you also need the concept of “STACK” data structure.

6. Storage classes in C

You must understand where your variables are getting stored and how long you can access them. In C, there are mainly four storage classes: automatic, static, external, and register. Each of them has different characteristics that you need to understand before implementing them. Here are a few things you need to know about them:

- Different storage areas,
- Keywords used to represent them,
- Default value of variables created under each storage class
- Scope of the storage classes
- Their lifetime
- Syntax
- Implementation in C program

7. Array

It is a collection of variables of similar data types and can be called using a common name. All C programmers need to learn arrays because no other data type can contain more than one variable, but an array can. For example, you want to store the first 10 odd numbers in one place. This isn’t possible with a char, int, or float, but an array can store as many variables as you want. Arrays can be either single-dimensional or multi-dimensional. You need to learn a few things about them, like

- Array declaration,
- Array initialization
- Accessing elements of an array
- Entering data into an array
- Reading/accessing data from an array
- Use of array in C programming

8. Strings

Strings are a one-dimensional array used to store a group of characters that ends with a null (‘\O’). C doesn’t have a String type to create or store string variables like several other programming languages. You must understand the string declaration syntax and the different ways of initialising a string. These string functions are bundled in the string.h library, and therefore, you must know how to define them in your program.

9. Pointers in C programming

Pointers are the variables in your C program that store a memory address. This address is the location of another variable stored in memory. There are multiple types of pointers, like null, void, wild, and dangling pointers. Each of them has a different purpose and function that you must know.