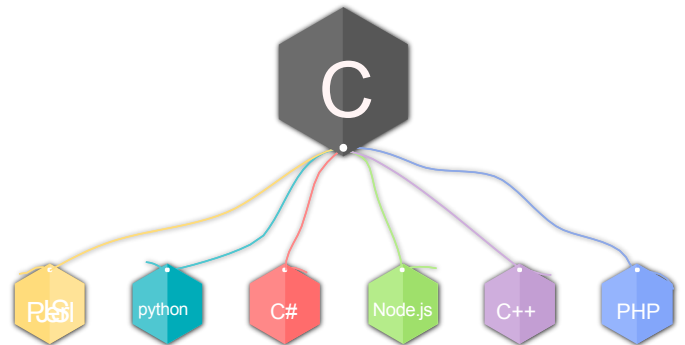


C Programming Syllabus MGTI Training Institute



ranjanss378@gmail.com



DETAILS

C programming language is a machine-independent programming language that is mainly used to create many types of applications and operating systems such as Windows, and other complicated programs such as the Oracle database, Git, Python interpreter, and games and is considered a programming foundation in the process of learning any other programming language.

Anyone and everyone can learn C Programming Language as it is easy to understand. Post learning C language aspiring job-seekers can look to learn more technologies coded on C language to further enhance their Resume

MGTI Institute

Contact - 9625418087

Introduction to Programming

- Fundamentals in C
- Operators and Expressions
- Data types
- Input-Output Library Functions
- Control statements
- Function
- Storage class
- Pointer
- Pointer and Function
- Array
- Pointer and array
- Array and function
- Dynamic memory allocation
- String
- String and function
- Command line arguments
- Preprocessor
- Structure
- Structure and function
- File Handling

SYLLABUS DETAILS

- Introduction to Programming
- Program and Programming
- Programming Languages
- Types of software's
- Operating Systems
- Dos commands
- Basic Linux commands and vi editor
- Compiler, Interpreter, Loader and Linker

Fundamentals in C

- History of 'C'
- A Simple C Program
- Program execution phases
- Backslash character constants
- Character set
- Constants
- Number systems
- Format specifiers

- Identifiers
- Keywords
- Variables
- Data Types
- Declaration of Variable
- Assigning Values to Variables
- Initialization
- Comments
- Const Qualifier
- Basic Structure of a 'C' program
- Programming Examples

Operators and Expressions

- Arithmetic operators
- Increment and decrement operators
- Relational operators
- Logical operators
- The bitwise operators
- The assignment operators
- The conditional operator
- The size of operator
- The comma operator
- Type casting operator
- Other operators
- Precedence and order of evaluation
- Programming Examples

Data types

- Modifiers
- Format specifiers
- Dealing with each data types
- Memory representation of each type
- Programming Examples

Input-Output Library Functions

- Unformatted I-O Functions
- Single Character Input-Output
- String Input-Output
- Formatted I-O Functions
- printf() Width Specifier

- scanf() Width Specifier
- Programming Examples

Control statements

- Conditional Control Statements
 - if
 - if-else
 - nested if-else
 - else-if ladder
 - Multiple Branching Control Statement
 - switch-case
- Loop Control Statements
 - while
 - do-while
 - for
 - Nested Loops
- Jump Control statements
 - break
 - continue
 - goto
 - exit
 - return
- Programming Examples

Function

- What is function?
- Why function?
- Advantages of using functions
- Function Prototype
- Defining a function
- Calling a function
- Return statement
- Types of functions
- Recursion
- Nested functions
- main() function
- Library Function
- Local and global variables
- Programming Examples

Storage class

- Types of storage class
- Scoping rules
- Dealing with all storage classes
- Programming Examples

Pointer

- Def of Pointer
- Declaration of Pointer Variables
- Assigning Address to Pointer Variables
- De-referencing Pointer Variables
- Pointer to Pointer
- Pointer Arithmetic
- Pointer comparisons
- D-reference and increment pointer
- Programming Examples

Pointer and Function

- Parameter Passing Techniques – call by value, call by address
- Using Pointers as Arguments Function Returning value
- Returning More than one value From A Function
- Functions Returning Address
- Function Returning Pointers
- Dangling pointer
- Pointer to a Function
- Calling A function through function pointer
- passing A function's address as an Argument to other function
- Functions with variable number of arguments
- Programming Examples

Array

- One dimensional arrays
- Declaration of 1D arrays
- Initialization of 1D arrays
- Accessing element of 1D arrays
- Reading and displaying elements

- Two dimensional arrays
- Declaration of 2D arrays
- Initialization of 2D arrays
- Accessing element of 2D arrays
- Reading and displaying elements
- Programming Examples

Pointer and Array

- Pointer and one dimensional arrays
- Subscripting pointer variables
- Pointer to an array
- Array of pointers
- Pointers and two dimensional arrays
- Subscripting pointer To an array
- Programming Examples

Array and Function

- 1D array and function
- Passing individual array elements to a function passing individual array elements address to a function
- passing whole 1d array to a function
- 2D array and function
- Passing individual array elements to a function
- Passing individual array elements address to a function
- passing whole 2d array to a function
- using arrays of function pointer
- Programming Examples

Dynamic memory allocation

- malloc()
- calloc()
- realloc()
- free()
- Core dump
- Memory leak
- Dynamic 1D and 2D Arrays
- Programming Examples

Strings

- strings versus character arrays
- Initializing strings
- Reading string
- Displaying string
- The %s format specifier
- The gets() and puts() functions
- string handling functions
- string pointers
- Two-dimensional character arrays or array of string
- array of pointers to strings
- Programming Examples

Command line arguments

- what is command prompt?
- why command line?
- What are command line arguments?
- Programs using command line

Preprocessor

- What is preprocessing?
- Macro expansions
- File inclusions
- Conditional compilation
- Programming Examples

Structure

- Why is structure used?
- What is structure?
- Advantages of structures
- Defining a Structure
- Declaration of Structure Variables
- Initialization of Structure Variables
- Accessing Structure Members
- Storage of Structures in Memory
- Size of Structures
- Reading and Displaying Structure Variables
- Assignment of Structure Variables

- Pointers to structures
- Array of structures
- Arrays within structures
- Nested structures
- Self-referential structures
- Programming Examples

Structure and Function

- Passing structure member to a function
- Passing structure variable to a function
- Passing structure variable address to a function
- Passing array of structure to a function
- Returning a structure variable from function
- Returning a structure variable address from function
- Returning structure variable from a function
- Programming Examples

Union and Enumeration and typedef

- What are unions?
- Structures versus unions
- Working with unions
- Initialising unions
- Advantages of unions
- enum keyword
- typedef keyword
- Programming Examples

File Handling

- Using files in C
- Buffer and streams
- Working with text files and Binary
- Files
- File operations using std. library and system calls
- File management I/O functions
- Random Access Files
- Programming Examples

Syllabus Updates : We keep adding/upgrading our syllabus every 6-12 months to accommodate for advancement. Some topics can be omitted from syllabus without prior notice as they might be redundant or not relevant to the batch being trained. You can request any Topic from your trainer.

MGTI COMPUTER INSTITUTE