C Training

A training pack for students & beginners

Dear Sir/Madam

Sub: To organize C Training in your college.

This is to bring to your kind notice that **POSITIVE QUADRANT TECHNOLOGIES LLP** is an Indian entity exploring itself in various sectors like Software Development , Augmented Reality , Virtual Reality , IoT , Simulation ,Games Development ,Mobile Applications,3D Modelling Development , Practical Educational Training, Professional Training, Corporate Training, Web & IT Services.

C Training conceptualized by **some top industry professionals** in association with **POSITIVE QUADRANT TECHNOLOGIES LLP**. It is going to be India's first & biggest workshop series based on this market flag bearer of cloud systems.

This workshop will also provide a platform where young engineers can mould their imagination into reality and feel the excitement first-hand. With this end in view, we extend our support and technical expertise to the young engineers of your College in the form of this workshop. We seek your cooperation and look forward towards a successful execution of this workshop in your college.

We are hoping that you will find this training really interesting for the students. If you have any queries, please get back to us anytime.

SYLLABUS

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

- –Dealing with all 45 operators
- –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
- –FAQ's

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
- –FAQ's

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
- –FAQ's

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
- –De-reference and increment pointer
- –pointer to const data
- –const pointer
- –const pointer to const data
- –Void pointer or Generic Pointer
- –Null pointer
- –wild 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

Up to 16 Hour’s

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
- FAQ's 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
- –FAQ's

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
- –FAQ's

Up to 30 Hour's

Dynamic memory allocation

- –malloc()
- –calloc()
- –realloc() .
- –free()
- –Core dump
- –Memory leak
- –Dynamic 1D and 2D Arrays
- –Programming Examples
- –FAQ's

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
- –FAQ's

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
- –The stringification(#)and token passing
- operator
- –(##) operators
- –Programming Examples

Up to 45 Hour’s

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
- –memory link(linked list)
- –Bit fields
- –Programming Examples
- –FAQ's

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
- –FAQ's

Union and Enumeration and typedef

- –What are unions?
- –Structures versus unions
- –Working with unions
- –Initializing unions
- –Advantages of unions
- –enum keyword
- –typedef keyword
- –Programming Examples
- –FAQ's

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
- –FAQ's

ODBC Programming

- –ODBC rules and regulation
- –Introduction to MYSQL and Oracle
- –Creating, inserting and retrieving
- records for different Data bases.
- –Programming Examples
- –FAQ's

Process and Threads

- –What is process & Threads
- –Use of fork, vfork
- –Daemon process
- –Programming Examples

Project

Up to 60 Hour’s