C Programming

SYLLABUS

COVERAGE

- 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
- ODBC Programming
- Process and threads
- Graphics
- Socket and Network programming
- Project

SYLLABUS IN 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

- 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
- FAQ's

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
- FAQ's

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
- FAQ'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'sPointer 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

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
- FAQ'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
- FAQ's

Graphics & Curses

- Graphics using Glade interface with GTK+
- Working with GTK Widgets, Event handling
- Developing Application Interface
- Programming Examples
- FAQ's

C ++

SYLLABUS

COVERAGE

- Introduction to Programming
- Fundamentals in C++
- Control statements
- Pointer array Reference
- Function
- Introduction to oops
- Classes and Objects
- Constructors and Destructors
- Operator Overloading
- Inheritance and Composition
- Polymorphism
- Exception handling
- Input / Output in C++: Streams
- File handling
- Working with String
- Command line arguments
- Namespace
- Templates
- Data Structures(introduction)
- STL
- Process and Threads
- Graphics
- WEB development
- Project

SYLLABUS IN DETAILS

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

Fundamentals in C++
- History of 'C++'
- Migrating from procedural oriented language
- to object oriented languages Program
- Keywords
- Variables
- Constants
- Data type
- Operators
- Manipulators and uses
- Basic Structure of a 'C++' program

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

Pointer array Reference
- pointer variable
- Reference variable/alias variables?
- Reference to Reference variable?
- Reference to array?
- Reference vs normal variable?
- Reference vs pointer variable?
- 1D and 2D Arrays
- What is dynamic memory allocation?
- The new and delete operator
- new vs malloc
- delete vs free
- Dynamic 1D and 2D Arrays

Function
- What is function ?
- Why function ?
- Advantages of using functions
- Function Prototype
- Defining a function
- Calling a function
- Actual and Formal Arguments
- Types of functions
- Parameter Passing Techniques
- Call by Value
- Call by Reference
- Call by Pointer
- Return statement
- Returning More than one value From A Function
- Return by value mechanism
- Return by pointer mechanism
- Return by reference mechanism
- Inline Functions
- Default Arguments
- Function Overloading
- Lambda function.
- Recursion

Introduction to oops
- c structure vs c++ structure
- c++ class vs c++ structure
- Class
- Object
- Encapsulation
- Abstraction
- Polymorphism
- Inheritance
- Message Passing

Classes and Objects
- Declaring / defining classes
- Data members and member functions
- Access specifiers : public and private and protected
- Creating objects of a class
- Pointers to object
- Implicit this pointer
- Static data members
- Static member functions
- Passing objects to a member function
- Returning objects from a member function
- Friend functions
- Friend classes
- Nested classes
- Local classes
- The const member functions
- The const objects
- Array of objects
- static objects
- What are inline functions?

Constructors and Destructors
- Defining Constructor
- Defining Destructor
- Comparing Constructor Member Function
- Default Constructor

- Argument Constructor
- Copy Constructor
- Constructor Overloading
- Default Argument in Constructor
- Anonymous object
- Private Constructor and Destructor
- Local vs Global object

Operator Overloading
- Need of Overloading
- Defining Operator Overloaded Function
- Operator Overloading Rules
- Overloading Binary Operators
- Overloading Binary Operators using Friend
- Overloading Other Operators
- Overloading Unary Operators
- Overloading Unary Operators using Friend

Inheritance and Composition
- What is inheritance?
- The is-a relationship
- Single Level Inheritance
- Multilevel Inheritance
- Multiple Inheritance
- Name ambiguities under multiple inheritance
- Hierarchical Inheritance
- Hybrid Inheritance
- Multipath Inheritance
- Why virtual base classes?
- Constructor and Destructor in Inheritance.
- Common constructor.
- Delegation
- What is composition?
- The has-a relationship

Polymorphism
- About polymorphism
- Compile time and runtime polymorphism
- Virtual functions.
- Pure virtual function and abstract base class.
- What is RTTI (Run-time Type Information)?
- VPTR and VTABLE.
- Difference between member Function Overloading and Overriding
- Object slicing.
- Constructor and virtual function.
- Virtual destructor.
- Destructor with virtual function.

Exception handling
- What is an exception?
- Throwing an exception
- Catching an exception
- Trying for an exception
- Order of catch blocks
- Catching all exceptions
- Nested try blocks
- Rethrowing an exception
- Exception specifications
- What is stack unwinding?
- Exceptions in ctors and dtors
- The unexpected() function
- The terminate() function
- The standard exceptions
- Creating our own exception classes

File handling
- Hierarchy of File Streams
- Using constructor method
- Using open and close member function method.
- Object as file stream reader and writer
- Both sequential and random file accessing mechanism.
- Different error handling mechanism in files

Input / Output in C++: Streams
- Hierarchy of I/O Streams
- Fundamental stream classes and objects
- Standard input and output functions
- Formatting flags and manipulators

Working With String
- Different C string handling library
- string handling using relational operator
- Different string handling function

Namespace
- Creating name space
- Using name space
- Nested namespace and anonymous namespace

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

Data Structures
- Introduction
- Single Linked List
- Circular Linked List
- Doubly Linked List
- Stacks
- Queues

Templates
- What is generic programming?
- Need of Template
- What are function templates?
- Argument deduction and template parameters
- Overloading function templates
- What are class templates?
- Specializations of class templates

STL
- STL Components
- Containers
- Iterators
- Algorithms
- Common container operations
- Vectors
- Deques
- Lists
- Sets and multisets
- Maps and multimaps
- Implementing reference semantics
- When to use which container?
- Special STL Containers
- Stacks
- Queues
- Priority Queues
- Bitsets
- STL Iterators
- Input iterators
- Output iterators
- Forward iterators
- Bidirectional iterators
- Random access iterators

Database operation
- What is database?
- SQL for relational database.
- About API connect to database.
- Database connectivity MySQL.
- Database manipulation using C++
- Process and Threads
- Graphics
- WEB development
- Web basics.(HTML, java script, CSS).