TRINITY Technologies

C Training Syllabus

Introduction

- About C
- The C Character Set

Structure of a C program

- Comments
- Variables, Data Types
- Identifiers
- Fundamental Data Types
- Declaration of Variables
- Scope of Variables
- Initialization of Variables
- C Keywords

A Quick Overview of C

- A Very Simple Program
- Compilation and Execution

Constant and Variable Types

- Variables
- Variable Names

- Global Variables
- Static Variables
- Constants
- Arrays

Data Types

- Integer long, short, signed and unsigned
- Float and Double
- Chars signed and unsigned

Operators

- Assignation (=)
- Arithmetic Operators (+, -, *, /, %)
- Compound assignation (+=, -=,*=, /+, %+, >>=, <<=, &=, ^=, |=)
- Increase and Decrease (++, --)
- Relational and Equality Operators (==,! =, >, <, <=, >=)
- Logical Operators (! &&, | |)
- Conditional Operators (?)
- Comma Operators (,)
- Bitwise Operators (&, |, ^, ~, >>, <<)
- Explicit type casting operator

- Sizeof ()
- Other Operator
- Precedence of Operators

Basic Input / Output

- The Standard Input Output File
- Character Input / Output
- getchar
- putchar
- printf
- scanf
- gets
- puts

Control Statements

The Decision Control Structure

- Decisions! Decisions!
- The *if* Statement
- The Real Thing
- Multiple Statements within *if*
- The *if-else* Statement
- Nested *if-elses*

- Forms of *if*
- Use of Logical Operators
- The *else if* Clause
- The ! Operator
- Hierarchy of Operators Revisited
- A Word of Caution
- The Conditional Operators

The Loop Control Structure

- Loops
- The *while* Loop
- Tips and Traps
- More Operators
- The *for* Loop
- Nesting of Loops
- Multiple Initialisations in the *for* Loop
- The Odd Loop
- The *break* Statement
- The *continue* Statement
- The *do-while* Loop

The Case Control Structure

- Decisions Using *switch*
- The Tips and Traps
- *switch* Versus *if-else* Ladder
- The *goto* Keyword

Arrays

- What are Arrays
- A Simple Program Using Array
- More on Arrays
- Array Initialization
- Bounds Checking
- Passing Array Elements to a Function
- Pointers and Arrays
- Passing an Entire Array to a Function
- The Real Thing
- Two Dimensional Arrays
- Initializing a 2-Dimensional Array
- Memory Map of a 2-Dimensional Array
- Pointers and 2-Dimensional Arrays
- Pointer to an Array
- Passing 2-D array to a Function

- Array of Pointers
- Three Dimensional Array

String Functions

- What are Strings
- More about Strings
- Pointers and Strings
- Standard Library String Functions
- Strlen()
- Strcmp()
- Strcpy()
- Strcat()
- Strrev()
- String sorting
- Other String Handling Functions
- Two-Dimensional Array of Characters
- Array of Pointers to Strings
- Limitation of Array of Pointers to Strings

Structures and Union in C

- Defining a Structure
- Why Use Structures

- Declaring a Structure
- Accessing Structure Elements
- How Structure Elements are Stored
- Array of Structures
- Additional Features of Structures
- Uses of Structures
- Structures as Function Arguments
- Further Uses of Structures
- Structures using array

Functions

- Scope of variables
- Declaring functions
- Functions with no type. The use of void
- Function with Arguments
- Function with Arguments and returns
- Arguments passed by value and by reference
- Recursion Function

Pointers

- Reference operator (&)
- Dereference operator (*)

- Declaring variables of pointer types
- Pointers and arrays
- Pointer initialization
- Pointer arithmetic
- Pointers to pointers
- Void pointers
- Null pointer
- Pointer to functions

Dynamic memory

- Sizeof()
- Malloc()
- Calloc()
- Free()
- Realloc()

Handling Files in C

- C File Handling - File Pointers
- Opening a file pointer using fopen()
- Closing a file using fclose()
- Text files
- Input and Output using file pointers

- Character Input and Output with Files
- Formatted Input Output with File Pointers
- Formatted Input Output with Strings
- Whole Line Input and Output using File Pointers
- Other File Handling Functions
- Input and Output using fread() and fwrite() function
- Input and Output using structures
- NULL, The Null Pointer or Character
- EOF, The End of File Marker
- Get and put stream pointers
- Tellg() and tellp()
- Seekg () and seekp ()
- Binary files
- Programs with Several Files
- Advantages of Using Several Files
- How to Divide a Program between Several Files
- Organisation of Data in each File
- Compiling Multi-File Programs
- Separate Compilation

- Using make with Multi-File Programs

C Preprocessor

- Using #define to Implement Constants
- Using #define to Create Functional Macros
- Using #undef
- Reading in Other Files using #include
- Conditional selection of code using #ifdef
- Using #ifdef for Different Computer Types
- Using #ifdef to Temporarily Remove Program Statements

Storage Class in C

- Automatic Storage Class
- Register Storage Class
- Static Storage Class
- External Storage Class

Miscellaneous Features

- Use of Enumerated Data types
- Renaming Data type with typedef

C Graphics

- Graphics.h

Internet Programming

- Network Communication
- Packets nd Sockets
- Protocole
- IP Address
- Port Number
- Byte Ordering
- Creation of Socket
- Sending Data using TCP/IP
- Sending Data using UDP

C++ Training Syllabus

Principles of Object-Oriented Programming

A Look at Procedure-Oriented Programming

Object-Oriented Programming Paradigm

Basic Concepts of Objected-Oriented Programming

Benefits of OOP

Object-Oriented Languages

Applications of OOP

Beginning with C++

What is C++

Application of C++

A Simple C++ Program

More C++ Statements

An Example with Class

Structure of C++ Program

Creating the Source File

Compiling and Linking

Tokens, Expressions and Control Structures

Introduction

Tokens

Keywords

Identifiers and Constants

Basic Data Types

User-Defined Data Types

Derived Data Types

Symbolic Constants

Type Compatibility

Declaration of Variables

Dynamic Initialization of Variables

Reference Variables

Operators in C++

Scope Resolution Operator

Member Dereferencing Operators

Memory Management Operators

Manipulators

Type Cast Operator

Expression and their Types

Special Assignment Expressions

Implicit Conversions

Operator Overloading

Operator Precedence

Control Structures

Functions in C++

Introduction

The Main Function

Function Prototyping

Call by Reference

Inline Function

Default Arguments

Const Arguments

Function Overloading

Friend and Virtual Functions

Math Library Functions

Classes and Objects

Introduction

C Structures Revisited

Specifying a Class

Defining Member Functions

A C++ Program with Class

Making an Outside Function Inline

Nesting of Member Functions

Private Member Functions

Array within a Class

Memory Allocation for Objects

Static Data Members

Static Member Functions

Arrays of Objects

Objects as Function Arguments

Friendly Functions

Returning Objects

Const Member Functions

Pointers to Members

Local Classes

Constructors and Destructors

Introduction

Constructors

Parameterized Constructors

Multiple Constructors in a Class

Constructors with Default Arguments

Dynamic Initialization of Objects

Copy Constructor

Dynamic Constructors

Constructing Two-dimensional Array

Const Objects

Destructors

Operators Overloading and Type Conversions

Introduction

Defining Operator Overloading

Overloading Unary Operators

Overloading Binary Operators

Overloading Binary Operators Using Friends

Manipulation of Strings Using Operators

Rules for Overloading Operators

Type Conversions

Inheritance: Extending Classes

Introduction

Defining Derived Classes

Single Inheritance

Making a Private Member Inheritance

Multilevel Inheritance

Multiple Inheritance

Hierarchical Inheritance

Hybrid Inheritance

Virtual Base Classes

Abstract Classes

Constructors in Derived Classes

Member Classes: Nesting of Classes

Pointers, Virtual Functions and Polymorphism

Introduction

Pointers

Pointers to Objects

This Pointer

Pointer to Derived Classes

Virtual Functions

Pure Virtual Functions

Managing Console I/O Operations

Introduction

C++ Streams

C++ Streams Classes

Unformatted I/O Operations

Formatted Console I/O Operations

Managing Output with Manipulators

Working with Files

Introduction

Classes for File Stream Operations

Opening and Closing a File

Detecting end-of-file

More about Open (): File Modes

File Pointers and Their Manipulations

Sequential Input and Output Operations

Updating a file: Random Access

Error Handling During File Operations

Command-line Arguments

Templates

Introduction/Class Templates

Class Templates with Multiple Parameters

Function Templates

Function Templates with Multiple Parameters

Overloading of Templates Functions

Member Function Templates

Non-Type Template Arguments

Exception Handling

Introduction

Basic of Exception Handling

Exception Handling Mechanism

Throwing Mechanism

Catching Mechanism

Rethrowing an Exception

Specifying Exceptions