Programming with C++

C++ is undoubtedly one of the most widely-used programming language for implementing object-oriented systems. The C++ language is based on the popular C language. The C++ Programming course provides thorough practical and theoretical coverage of the C++ language for the experienced application programmer who has little or no recent C++ experience.

This course helps eliminate misconceptions and poor programming practices that can cause so many problems, by focusing on features of the language and standard library that enforce good practice and encourage clear and robust code.

This is a highly practical course which uses a mix of tuition and practical sessions for each technical chapter designed to reinforce the C++ syntax and object-oriented programming techniques covered in the course.

Delegates will learn how to

- Define and use classes
- Use fundamental and composite data types
- Understand the key concepts and vocabulary of object orientation
- Write class member functions
- Use pointers and dynamic memory
- Use constructors and destructors
- Write code that is efficient and robust
- Build new classes from other classes using aggregation and association
- Build new classes from other classes using inheritance
- Use container classes, including template classes
- Use operator overloading
- Design and write code with polymorphic behavior

Course Introduction

- Course Prerequisites
- Course Objectives
- Course Delivery
- Course Practicals
- Course Structure

C ++ Programs

- Key features of C++
- Identifiers and keywords
- Simple declarations, expressions and statements
- Basic I/O
- Layout
- Guidelines

Fundamental Data Types

- Built-in types
- Integer numbers
- Floating Point numbers
- Characters
- Booleans
- Assignment
- Compound Assignment
- Increment and Decrement
- Defining constants
- Type conversions

Composite Data Types

- Defining and using enumerations
- Built-in arrays and their limitations
- Using the vector class
- Built-in strings as character arrays
- Using the string class
- Defining and using structures

Control Flow

- Simple and compound statements
- Selection with if else and switch statements
- Conditional expressions
- Looping with while and for statements

Functions

- Declaring, calling and defining functions
- Overloading
- Default arguments
- Scope issues
- Pass by copy
- Pass by reference
- Inline functions
- Header files and source files
- Pitfalls and guidelines

Object Concepts

- Object behaviour
- Object state
- Object identity, Object-oriented programming
- Classes
- Encapsulation

Using Classes

- Associating functionality with data
- Class definitions
- Public and private

- Queries functions and modifier functions
- Struct vs class

Pointers

- Concepts and syntax
- Pointers to structured types
- Pointers for encapsulated objects
- Null pointers
- Pointers vs. references

Implementing Classes

- Defining member functions
- Object identity
- The this pointer
- Initialisation
- Constructors
- Default constructors
- Member Initialisation
- Scope issues
- Inlining member functions

Operator Functions

- Operators as functions
- Global operators
- Member operators
- I/O stream operators
- Pitfalls and guidelines

Object Relationships

- Associations and their implementation
- Compositions and their implementation
- Navigation
- Delegation
- Multiplicity

Dynamic Memory

- The need for dynamic memory
- Dynamic objects
- Using new and delete
- Dynamic arrays
- Using new[] and delete[]
- Destructors

More Pointers

- Pointers and arrays
- Pointer arithmetic
- Pointers as array iterators
- Pointers and const

- Pointers vs. references

Containers

- Container concepts and classification
- Template classes
- Standard containers
- Vector
- List
- Iterators
- Template functions
- Algorithms

Copying

- Copy construction
- Copy assignment
- Compiler generated copy behaviour
- Problems
- Solutions

Delegates who are relatively new to programming or who do not have experience in a modern programming environment, for instance on mainframe systems, should first attend the Programming Foundation