

Introduction to C++ Programming

Course Summary

Description

The C++ programming language is a widely used powerful tool for producing modern object-oriented programs. This course guides experienced programmers through the complexities of writing and maintaining object-oriented programs in C++. It is fully up to date with the latest ISO standard for C++ and includes coverage of the important Standard Template Library.

Objectives

At the completion of this course, the student will be able to:

- Write, maintain and debug C++ programs
- Use dynamic memory
- Make effective use of the Standard Library including the Standard Template Library
- Write object-oriented code using encapsulation, inheritance and polymorphism

Topics

- C++ Program Components
- Data and Operators
- Structured Data Types
- Control Flow
- Functions
- Arrays and Vectors
- Classes
- Pointers
- Dynamic Memory
- Object Relationships
- Operator Overloading
- Streams
- STL Containers
- Iterators
- Inheritance
- Polymorphism
- Advanced Features

Audience

This course is designed for experienced programmers who need to write or maintain object-oriented C++ programs.

Prerequisites

It is assumed that students attending this course are experienced in a block-structured language such as Pascal, modern variants of Basic, Smalltalk, Ada etc. They should be familiar with concepts such as functions or procedures, control flow, arrays and structures or records. Knowledge of C is neither assumed nor required.

Duration

Five days

Due to the nature of this material, this document refers to numerous hardware and software products by their trade names. References to other companies and their products are for informational purposes only, and all trademarks are the properties of their respective companies. It is not the intent of ProTech Professional Technical Services, Inc. to use any of these names generically

Introduction to C++ Programming

Course Outline

- I. C++ Program Components**
 - A. Basic program components
 - B. Functions
 - C. Statements
 - D. File inclusion
 - E. Input and output
 - F. Keywords
 - G. Variables
 - H. Operators

- II. Data and Operators**
 - A. Identifiers
 - B. Fundamental data types and their operators
 - C. Constants
 - D. Operator precedence
 - E. Conversions
 - F. Casting
 - G. Scope

- III. Structured Data Types**
 - A. Enumerations
 - B. Data structures
 - C. Aggregation
 - D. Standard Library string class.

- IV. Control Flow**
 - A. Boolean operators
 - B. Conditional selection
 - C. Iteration

- V. Functions**
 - A. Functional modularity
 - B. Passing data in and out of functions
 - C. Pass by value and reference
 - D. Member functions
 - E. Optimization with inline

- VI. Arrays and Vectors**
 - A. Basic containers
 - B. Arrays and the standard template library vector class
 - C. Access through subscripting and member functions
 - D. C type strings.

Due to the nature of this material, this document refers to numerous hardware and software products by their trade names. References to other companies and their products are for informational purposes only, and all trademarks are the properties of their respective companies. It is not the intent of ProTech Professional Technical Services, Inc. to use any of these names generically

VII. Classes

- A. Object-orientation and classes
- B. Encapsulation
- C. Automatic construction
- D. Copying and conversions

VIII. Pointers

- A. Concept of indirection
- B. Null pointers
- C. Using pointers with functions and arrays
- D. Pointer arithmetic

IX. Dynamic Memory

- A. Accessing the heap or freestore with new and delete
- B. Dynamic arrays
- C. Automatic destruction

X. Object Relationships

- A. Aggregation and association
- B. Delegation, managing custody of dynamic memory
- C. Copying custodial objects

XI. Operator Overloading

- A. Class operators
- B. Overloading with globals or members
- C. Lvalue operators
- D. Copy assignment

XII. Streams

- A. Working with the Standard Streams
- B. File streams
- C. Manipulators
- D. String streams

XIII. STL Containers

- A. Introducing templates and the main Standard Template Library container classes
 - 1. vector, deque, list, set and map
- B. Guidelines on choosing the right container for the job

Introduction to C++ Programming

Course Outline (Cont'd)

XIV. Iterators

- A. Using Iterators to access the elements of Standard Template Library containers
- B. Introducing the standard algorithms

XV. Inheritance

- A. Object-orientation and inheritance
- B. Creating and using derived classes
- C. Vertical delegation
- D. Standard conversions

XVI. Polymorphism

- A. Declaring and using virtual functions and the need for virtual destructors
- B. Pure virtual functions and abstract classes

XVII. Advanced Features

- A. Overview of some advanced features
- B. Exception handling
- C. Namespaces
- D. Multiple inheritance
- E. Smart pointers
- F. Template functions
- G. Extending the STL