

## C++ Programming for C Programmers

**Duration:** 4 days (*Face-to-Face & Remote-Live*), or 28 Hours (*On-Demand*)

**Price:** \$2095 (*Face-to-Face & Remote-Live*), or \$1495 (*On-Demand*)

**Discounts:** We offer multiple discount options. [Click here](#) for more information.

**Delivery Options:** Attend face-to-face in the classroom, [remote-live](#) or [on-demand training](#).

### Students Will Learn

- Writing procedural programs using C++
- Using `private`, `public` and `protected` keywords to control access to class members
- Defining a class in C++
- Writing constructors and destructors
- Writing classes with `const` and `static` class members
- Overloading operators
- Implementing polymorphic methods in programs
- Writing programs using file I/O and string streams
- Using manipulators and stream flags
- to format output
- Using the keyword `template` to write generic functions and classes
- Writing programs that use generic classes and functions
- Writing programs that use algorithms and containers of the Standard Library
- Using algorithms and containers of the Standard Library to manipulate string data
- Using `try()` blocks to trap exceptions
- Using `catch()` blocks to handle exceptions
- Defining exceptions and using `throw` to trigger them

### Course Description

This hands on C++ training course presents a thorough introduction to object-oriented programming in C++ for experienced C programmers. The central concepts of C++ syntax and style are taught in the context of using object-oriented methods to achieve reusability, adaptability and reliability. Emphasis is placed on the features of C++ that support abstract data types, inheritance, and polymorphism. Students will learn to apply the process of data abstraction and class design. Practical aspects of C++ programming including efficiency, performance, testing, and reliability considerations are stressed throughout. Comprehensive hands on exercises are integrated throughout to reinforce learning and develop real competency.

## Course Prerequisites

Prior programming experience with C.

## Course Overview

### Moving from C to C++

- New Compiler Directives
- Stream Console I/O
- Explicit Operators
- Standard Libraries
- Data Control Capabilities

### Functions

- Function Prototypes and Type Checking
- Default Function Data Types
- Function Overloading
- Problems with Function Overloading
- Name Resolution
- Promotions and Conversions
- Call by Value
- Reference Declarations
- Call-by-Reference and Reference Types
- References in Function Return
- Constant Argument Types
- Conversion of Parameters Using Default Initializers
- Providing Default Arguments
- Inline Functions

### Dynamic Memory Management

- Advantages of Dynamic Memory Allocation
- Static, Automatic, and Heap Memory
- Free Store Allocation with `new` and `delete`
- Handling Memory Allocation Errors

### Inheritance

- Inheritance and Reuse
- Composition vs. Inheritance
- Inheritance: Centralized Code

### Handling Data

- New Declaration Features
- Initialization and Assignment
- Enumerated Types
- The `bool` Type
- Constant Storage
- Pointers to Constant Storage
- Constant Pointers
- References
- Constant Reference Arguments
- Volatile Data
- Global Data

### Creating and Using Objects

- Creating Automatic Objects
- Creating Dynamic Objects
- Calling Object Methods
- Constructors
- Initializing Member consts
- \_INITIALIZER List Syntax
- Allocating Resources in Constructor
- Destructors
- Block and Function Scope
- File and Global Scope
- Class Scope
- Scope Resolution Operator `::`
- Using Objects as Arguments
- Objects as Function Return Values
- Constant Methods
- Containment Relationships

### Controlling Object Creation

- Object Copying and Copy Constructor
- Automatic Copy Constructor
- Conversion Constructor

### Streaming I/O

- Streams and the `iostream` Library
- Built-in Stream Objects
- Stream Manipulators

- Inheritance: Maintenance and Revision
  - Public, Private and Protected Members
  - Redefining Behavior in Derived Classes
  - Designing Extensible Software Systems
- Syntax for Public Inheritance
- Use of Common Pointers
- Constructors and Initialization
- Inherited Copy Constructors
- Destructors and Inheritance
- Public, Protected, Private Inheritance
- Stream Methods
- Input/Output Operators
- Character Input
- String Streams
- Formatted I/O
- File Stream I/O
- Overloading Stream Operators
- Persistent Objects

## Introduction to Object Concepts

- The Object Programming Paradigm
- Object-Oriented Programming Definitions
- Information Hiding and Encapsulation
- Separating Interface and Implementation
- Classes and Instances of Objects
- Overloaded Objects and Polymorphism

## Strings in C++

- Character Strings
- The String Class
- Operators on Strings
- Member Functions of the String Class

## C++ Program Structure

- Organizing C++ Source Files
- Integrating C and C++ Projects
- Using C in C++

## Polymorphism in C++

- Definition of Polymorphism
- Calling Overridden Methods
- Upcasting
- Accessing Overridden Methods
- Virtual Methods and Dynamic Binding
- Virtual Destructors
- Abstract Base Classes and Pure Virtual Methods

## Declaring and Defining Classes

- Components of a Class
- Class Structure

## Templates

- Purpose of Template Classes
- Constants in Templates
- Templates and Inheritance
- Container Classes
- Use of Libraries

## Exceptions

- Types of Exceptions
- Trapping and Handling Exceptions
- Triggering Exceptions
- Handling Memory Allocation Errors

## Reliability Considerations in C++ Projects

- Function Prototypes
- Strong Type Checking
- Constant Types
- C++ Access Control Techniques

## Multiple Inheritance

- Derivation from Multiple Base Classes
- Base Class Ambiguities
- Virtual Inheritance
  - Virtual Base Classes
  - Virtual Base Class Information

## Operator Overloading

- Advantages and Pitfalls of Overloading
- Member Operator Syntax and

- Class Declaration Syntax
- Member Data
- Built-in Operations
- Constructors and Initialization
- Initialization vs. Assignment
- Class Type Members
- Member Functions and Member Accessibility
- Inline Member Functions
- Friend Functions
- Static Members
- Modifying Access with a Friend Class

#### Examples

- Class Assignment Operators
- Class Equality Operators
- Non-Member Operator Overloading
- Member and Non-Member Operator Functions
- Operator Precedence
- The this Pointer
- Overloading the Assignment Operator
- Overloading Caveats

### **The Standard Template Library**

- STL Containers
- Parameters Used in Container Classes
- The Vector Class
- STL Algorithms
- Use of Libraries

Hands On Technology Transfer  
The Best Way to Transfer Technology Skills

1 Village Square, Suite 8  
14 Fletcher Street  
Chelmsford, MA 01824

Copyright © 2021 Hands On Technology Transfer, Inc.