SHI | TRAINING & ADOPTION

Course Outline

Introduction to C++ Programming | C++ Essentials
Course TTCP2100: 4 days Instructor Led

About this course

Introduction to C++ Programming / C++ Essentials is a skills-focused, hands-on C++ training course geared for experienced programmers who need to learn C++ coupled with sounds coding skills and best practices for OO development. Students will leave this course armed with the required skills to put foundation-level C++ programming skills right to work in a practical environment.

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.

Audience profile

This is an introductory-level C++ programming course designed for developers with experience programming in C or other languages. Practical hands-on prior programming experience and knowledge is required.

NOTE: This course is for experienced developers. Students new to Programming should consider our Basic C++ Programming for Non-Programmers, which combines and introduction to programming with basic C++ coding skills.

At course completion

After completing this course, students will be able to:

- 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
- Apply object-oriented design techniques to real-world programming problems
- Using algorithms and containers of the Standard Library to manipulate string data
- Understand how C++ protects the programmer from implementation changes in other modules of an application
- Using try() blocks to trap exceptions
- Using catch() blocks to handle exceptions
- Defining exceptions and using throw to trigger them

Course Outline

SHI | TRAINING & ADOPTION

Course Outline

1. Moving from C to C++ (Optional)

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

2. 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

3.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

4.Operator Overloading

- Advantages and Pitfalls of Overloading
- Member Operator Syntax and Examples
- Class Assignment Operators
- Class Equality Operators
- Non-Member Operator Overloading
- Member and Non-Member Operator Functions

Course Outline

- Operator Precedence
- This Pointer
- Overloading the Assignment Operator
- Overloading Caveats

5. 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

6. Dynamic Memory Management

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

7.Controlling Object Creation

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

8.Streaming I/O

- Streams and the iostream Library
- Built-in Stream Objects
- Stream Manipulators
- Stream Methods
- Input/Output Operators
- Character Input
- String Streams
- Formatted I/O

Course Outline

- File Stream I/O
- Overloading Stream Operators
- Persistent Objects

9. Introduction to Object Concepts

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

10. Declaring and Defining Classes

- Components of a Class
- Class Structure
- 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

11. Templates

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

12. Strings in C++

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

13. Inheritance

- Inheritance and Reuse
- Composition vs. Inheritance

Course Outline

- Inheritance: Centralized Code
- 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

14. Exceptions

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

15. C++ Program Structure

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

16. Reliability Considerations in C++ Projects

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

17. 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

18. Multiple Inheritance

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

Course Outline

- Virtual Base Class Information

19. The Standard Template Library

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