Introduction to C++ 20 Programming Essentials

Duration: *5 Day(s)*

Course Overview

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.

NOTE: This course is for experienced developers. Students new to Programming should consider our TTCP2000 Introduction to Programming and C++ Basics for Non-Programmers, which combines and introduction to programming with basic C++ coding skills.

Review this course online at https://www.alta3.com/courses/TTCP2100

Objectives

- Design classes using C++ with proper access control and constructors.
- Implement polymorphic methods and handle exceptions effectively.
- Utilize the Standard Library for string manipulation and algorithms.
- Apply object-oriented design techniques to programming challenges.

Who Should Attend

- Experienced C programmers
- Developers familiar with other languages
- Software engineers
- IT professionals seeking C++ expertise

Prerequisites

Practical hands-on prior programming experience and knowledge is required.

Course Outline

Getting Started

1. Overview
2. Using the development environment
3. C++ file organization and tools

Handling Data

4. Primitive Types
5. Initialization and Assignment
6. Const
7. Pointers
8. Constant Pointers
9. References
10. Constant Reference Arguments
11. Scope

Functions

12. Function Prototypes and Type Checking
13. Function Overloading
14. Name Resolution
15. Call by Value
16. Call-by-Reference and Reference Types
17. References in Function Return
18. Constant Argument Types
19. Providing Default Arguments
20. Inline Functions

Declaring and Defining Classes

21. Components of a Class
22. Class Structure
23. Class Declaration Syntax
24. Member Data
25. Built-in Operations
26. Constructors and Initialization
27. Initialization vs. Assignment
28. Class Type Members
29. Member Functions and Member Accessibility
30. Inline Member Functions
31. Friend Functions
32. Static Members
33. Modifying Access with a Friend Class

Creating and Using Objects

34. Creating Automatic Objects
35. Creating Dynamic Objects
36. Calling Object Methods

37. Constructors
38. Initializing Member consts
39. Initializer List Syntax
40. Allocating Resources in Constructor
41. Destructors
42. Scope Resolution Operator ::
43. Using Objects as Arguments
44. Objects as Function Return Values
45. Constant Methods
46. Containment Relationships

Controlling Object Creation

47. Object Copying and Copy Constructor
48. Automatic Copy Constructor

Dynamic Memory Management

49. Static, Automatic, and Heap Memory
50. Free Store Allocation with new and delete
51. Handling Memory Allocation Errors

Strings in C++

52. Character Strings
53. The String Class
54. Operators on Strings
55. Member Functions of the String Class

Operator Overloading

56. Member Operator Syntax and Examples
57. Class Assignment Operators
58. Class Equality Operators
59. Non-Member Operator Overloading
60. Member and Non-Member Operator Functions
61. Operator Precedence
62. This Pointer
63. Overloading the Assignment Operator
64. Conversion Operators
65. Constructor Conversion Operator
66. Explicit vs Implicit conversion

Streaming I/O

67. Streams and the iostream Library
68. Built-in Stream Objects
69. Stream Manipulators
70. Stream Methods
71. Input/Output Operators
72. Character Input
73. String Streams
74. Formatted I/O
75. File Stream I/O
76. Overloading Stream Operators
77. Persistent Objects

Templates

78. Purpose of Template Classes
79. Constants in Templates
80. Templates and Inheritance
81. Container Classes
82. Use of Libraries

Inheritance

83. Inheritance and Reuse
84. Composition vs. Inheritance
85. Syntax for Public Inheritance
86. Use of Common Pointers
87. Constructors and Initialization
88. Inherited Copy Constructors
89. Destructors and Inheritance

Polymorphism in C++

90. Definition of Polymorphism
91. Calling Overridden Methods
92. Upcasting
93. Accessing Overridden Methods
94. Virtual Methods and Dynamic Binding
95. Virtual Destructors
96. Abstract Base Classes and Pure Virtual Methods

Exceptions

97. Types of Exceptions

98. Trapping and Handling Exceptions
99. Triggering Exceptions
100. Handling Memory Allocation Errors

The Standard Library

101. Survey of the library
102. Containers
103. Algorithms
104. Numerics
105. Date & Time