

Course Description: This course provides students with a comprehensive study of the C++ programming language while teaching those parts of C relevant to C++. Classroom lectures are supplemented with many hands-on exercises, which stress the following C++ topics: data abstraction, class design, operator overloading, inheritance, polymorphism and I/O.

Who Should Attend: This course is designed primarily for those Cobol, Pascal, and Fortran programmers who wish to learn C+ without having to partake in a prior C Language programming course.

Prerequisites: Experience with a programming language or an assembly language is required.

Benefits of Attendance: Upon completion of this course, students will be able to:

- Use correct object oriented terminology.
- Compare and choose between the object and the procedural approach to writing software.
- Compare and select the appropriate I/O model from either C or C++.
- Define and use classes in a C++ program.
- Select the proper class protection mechanism.
- Create and use abstract data types.
- Implement operator overloading in user defined and off the shelf classes.
- Derive classes using inheritance in C++.
- Implement polymorphism by using virtual functions in a C++ program.
- Utilize the modular features of the C and C++ language.

Course Outline:

Introduction

Background
Environmental considerations
A Sample C program
Variables and data types
Arrays
Components of a C program
C operators
Control structures
Functions
Function prototypes
Simple I/O

More I/O In C

The printf function
The scanf function
The preprocessor
Conditional compilation
Avoiding multiple inclusions for the same file

Aggregates In C

Data types revisited
Aggregate types
Arrays
Structures
Structures and functions
Bit fields
Enumeration types

Pointers In C

Fundamental Concepts
Pointer operations
Using pointers to alter a function argument
Using pointers for array traversal
Pointer arithmetic
Sending an array to a function
Pointers vs arrays
Sending an aggregate to a function
Summary of the uses of pointers

Perspective

The software crisis
Building software has been difficult
Design techniques
Large Software Systems
Roots of Object Orientation
What is OO programming?
C++ and OO programming
Why C++?
Features of C++
Pros and Cons of C++

The Language Of Object Orientation

What is an object?

What is a class?

Encapsulation
Data hiding
The public Interface
Relationships among Classes
Inheritance
Polymorphism
Object-Oriented Design

C vs C++

Comments
Namespaces
Performing Simple Output
Performing Simple Input
Definitions near to first Use
Function prototypes
The inline specifier
const
Structure Members
The reference type
Overloading function names
Default parameters
Scope resolution operator
Aggregates
Operators new and delete
The bool Data Type
The string Data Type

Fundamentals Of Classes

Data types
Abstract data types
Using the class concept
How to define a class
public and private access levels
Using class objects like a built in type
scope
scope resolution operator
Constructors
Member Initialization Lists
Destructors
Array of Objects
Pointers
The this pointer
Passing Objects to Functions
Returning Objects from Functions
Static class members

Operator Overloading

Introduction
Rules for Operator Overloading
Rationale for Operator Overloading
Overloading Member Functions
Overloading Non-Member Functions

friend functions

The Copy Constructor
Overloading the Assignment Operator
Overloading []
Overloading increment and decrement operators
const Objects & const references

Composition Of Classes

Relationships
Composition of Classes
The Point class
The Line class
Member Initialization Lists
An Application w/ composition
The Copy Constructor under Composition
Operator= under Composition

Inheritance

Introduction
Inheritance public base classes
Inheritance w/ public base classes
Member Initialization Lists
What isn't inherited
Assignments between base and derived Objects
Compile Time Binding vs. Run Time Binding
virtual functions
Polymorphism
virtual destructors
Pure virtual functions
Abstract base classes
An extended inheritance example

I/O In C++

The iostream library
Pre-defined streams
operator<<
Overloading << for User-Defined Classes
Overloading >> for User-Defined Classes
Manipulators
Stream states
Formatted i/o
Disk files
Internal transmission of data
Reading & Writing Objects

Advanced Topics

Template Functions
Template Classes
Multiple Inheritance
User-Defined Conversions
Data Structures
Iterators
Exceptions