C++ PROGRAMMING FOR NON C PROGRAMMERS

General:

This course provides experienced programmers with the ability to utilize the structure and syntax of the object oriented C++ programming language for commercial and systems programming applications. The student is prepared to code, test, and execute C++ programs making use of the facilities provided by the language. Previous knowledge of the C Programming Language is not required.

Objectives:

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

- Discuss the basic structure and syntax of the C++ Programming Language.
- Code object oriented C++ programs using all available control structures.
- Compile and execute C++ programs in various environments.
- Handle input/output processing using the standard C++ stream classes.
- Define encapsulation, inheritance, and polymorphism.
- Declare and use basic and derived data types.
- Effectively use arrays, pointers, and references within a C++ program.
- Create data abstractions through the use of classes.
- Use operator and function overloading.
- Share and restrict object members.
- Define and use constructors and destructors.
- Dynamically create and destroy space.
- Increase software reusability through inheritance.
- Use dynamic binding and virtual functions.

Audience:

Technical Users, Applications Programmers, and Systems Programmers.

Prerequisites:

Experience and working knowledge of another high level programming language.
Knowledge of one of the program editors for the development environment used for the exercise sessions.
An understanding of object oriented programming concepts is helpful.

Duration:

Five (5) days including classroom lecture and lab sessions.

PanTech
908-281-6759

C++ PROGRAMMING FOR NON C PROGRAMMERS
COURSE OUTLINE

I. **INTRODUCTION**

- A. Features of C++
- B. History and Origin of C++
- C. Advantages of C++
- D. C++ Compilation Process
- E. Basic Functions

II. **C++ BASICS**

- A. Comments
- B. Identifier Naming Rules
- C. Reserved Keywords and Modifiers
- D. Basic Data Representation
- E. Constants and Variables
- F. Input/Output Streams

III. **EXPRESSIONS AND OPERATORS**

- A. Expressions
- B. Basic Operators
  1. Assignment
  2. Arithmetic
  3. Compound Assignment
- C. Operator Precedence and Associativity

IV. **STATEMENTS**

- A. Flow Control
  1. `if`
  2. `switch`
  3. `break`
  4. `continue`
  5. `while`
  6. `do-while`
  7. `for`
  8. `goto`
- B. More Operators
  1. Relational and Equality
  2. Logical
  3. Conditional

V. **FUNCTIONS**

- A. Function Definition
- B. Function Declaration (prototype)
- C. Reference Parameters
- D. Default Argument Values
- E. Function Overloading
- F. Inline Functions
- G. Type Safe Linkage
- H. Name Mangling
- I. Combining C and C++ Functions

VI. **ARRAYS AND STRINGS**

- A. Defining and Using Arrays
- B. Two dimensional Arrays
- C. Processing Character Strings
- D. Passing Arrays to Functions

VII. **POINTERS**

- A. Defining a Pointer
- B. Declaring and Using Pointers
- C. Functions and Pointers
- D. Pointers and Arrays

VIII. **POINTERS REVISITED**

- A. Brief Review of Pointers
- B. Arrays of Pointers
- C. Accessing Command Line Arguments

IX. **OBJECT ORIENTED CONCEPTS**

- A. Features of Object Oriented Languages
- B. Procedural vs Object Oriented
- C. Encapsulation
- D. Inheritance
- E. Polymorphism
- F. Effects of OO Approach

X. **CLASSES AND ENCAPSULATION**

- A. Definition of Class
- B. Class Data Members
- C. Class Member Functions
- D. Private vs Public Members

PanTech
908-281-6759

XI. **CONSTRUCTORS AND DESTRUCTORS**

A. Class Constructors
   1. Constructors with Parameters
   2. Overloaded Constructors
   3. Internal and External Constructors
B. Class Destructors
C. Dynamic Memory Allocation
   1. `new` Operator
   2. `delete` Operator
D. Pointers to Classes
E. `this` Pointer
F. Function and Class Friends
G. Copy Constructors
H. Avoiding Memory Leaks

XII. **I/O STREAMS**

A. Standard I/O Streams
B. Reading Input with `cin`
C. Writing Output with `cout`
D. Writing Errors with `cerr`
E. Other I/O Class Member Functions
   1. `get`
   2. `put`
   3. `write`
F. Simple File I/O
   1. State Checking Member Functions
   2. Opening File Streams
   3. Closing File Streams
   4. Reading and Writing Files

XIII. **OVERLOADING OPERATORS**

A. Valid Operators
B. Overloading Operators
C. Overloading Binary Operators
D. Overloading Unary Operators
E. Friend Operator Functions
F. Non-Member Operator Functions

XIV. **INHERITANCE**

A. Software Reusability
B. Concept and Terminology
C. Uses and Advantages
D. Base and Derived Classes
E. Class Access Modifiers
   1. `public`
   2. `private`
   3. `protected`
F. Single Inheritance
G. Multiple Inheritance
H. Constructors and Destructors
I. Containment

XV. **POLYMORPHISM**

A. Polymorphism
B. Types of Binding
   1. Static
   2. Dynamic
C. Virtual Functions
D. Rules for Dynamic Binding
E. Pure Virtual Functions