ADVANCED C LANGUAGE PROGRAMMING

General:

This course provides experienced C Language Programmers with additional insights into the language. The student is given an in-depth understanding of the use of pointers, arrays, and structures in C Language, and is prepared to use these language facilities to write readable, portable, and efficient C Language Programs.

Objectives:

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

- Evaluate simple and complex expressions
- Effectively use expressions involving side effects
- Describe and use all C Language basic and derived data types
- Describe the attributes and results of all C Language operators
- Explain the scope and storage class of C Language objects
- Declare and use pointer data types and multi-levels of indirection
- Describe pointer arithmetic and use it to access array elements
- Declare and use single and multi-dimensional arrays
- Declare and use pointers to arrays and arrays of pointers
- Describe the difference between an array of pointers and a multi-dimensional array
- Reference single and multi-dimensional array elements using both pointers and array subscripting
- Declare and use structure data types including pointers to structures
- Declare and use functions returning pointers and structures
- Declare and use pointers to functions
- Dynamically allocate space and write functions to perform linked list processing

Audience:

Experienced C Language Applications Programmers, and Systems Programmers.

Prerequisites:

At least three (3) months of C Language Programming experience.

Duration:

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

PanTech
908-281-6750

ADVANCED C LANGUAGE PROGRAMMING
COURSE OUTLINE

I. **EXPRESSIONS, OBJECTS, AND LVALUES**

- A. Evaluating Expressions
- B. Expression Side Effects
- C. Objects and Data Types
- D. Lvalues and Constants

II. **DATA TYPES, CONSTANTS, AND VARIABLES**

- A. Basic Data Types
- B. Constant Data Types
- C. Variable Data Types
- D. Derived Data Types
- E. Data Type Renaming - `typedef`

III. **SCOPE AND STORAGE CLASS**

- A. Internal Scope
- B. External Scope
- C. Extending Scope
- D. Storage Class
  1. Automatic
  2. External
  3. Static
  4. Register

IV. **OPERATORS AND OPERANDS**

- A. Operator Attributes
  1. Operator Action
  2. Number of Operands
  3. Precedence Level
  4. Associativity Direction
- B. Operand Attributes
  1. Data Type Restrictions
  2. Lvalue Requirements
- C. Expression Evaluation Result
  1. Data Type
  2. Value
  3. Lvalue
  4. Side Effects

V. **ARRAYS AND POINTERS**

- A. Multi Dimensional Arrays
- B. Pointer - Array Relationship
- C. Pointer Arithmetic
- D. Valid Pointer Operations
- E. Multiple Pointers to an Array
- F. Array Subscripting vs. Pointers
- G. Arrays of Pointers
- H. Multiple Levels of Indirection

VI. **STRUCTURES AND POINTERS**

- A. Structure Review
  1. Structure Definition
  2. Structure Declaration
- B. Structure Holes
- C. Nested Structures
- D. Arrays of Structures
- E. Pointers to Structures
- F. Structures Containing Pointers
- G. Structure Variations
  1. Unions
  2. Bit Fields
  3. Enumerations

VII. **FUNCTIONS AND POINTERS**

A. Function Declaration
   1. Old Style
   2. Function Prototypes (ANSI)

B. Function Arguments and Parameters
   1. Basic Data Types
   2. Pointers to Basic Data Types
   3. Single and Multi-Dimensional Arrays
   4. Structures and Pointers to Structures

C. Function Return Types
   1. `void` Data Type Functions
   2. Basic Data Type Functions
   3. Pointer to Data Type Functions
   4. Structure Data Type Functions

D. Pointers to Functions
   1. Declaring Pointers to Functions
   2. Pointers to Functions as Formal Parameter
   3. Arrays of Pointers to Functions
   4. Functions Returning Pointers to Functions

VIII. **DYNAMIC MEMORY ALLOCATION & LINKED LISTS**

A. Dynamic Memory Allocation
   1. Dynamic Memory Allocation Concepts
   2. Library Function: `malloc`
   3. Library Function: `free`

B. Linked Lists
   1. Linked List Concepts
   2. Linear Linked List
   3. Bi-directional Linked List
   4. Linked List Conclusions