PROGRAMMING IN C LANGUAGE

General:

This course provides experienced programmers with the ability to utilize the structure and syntax of the C Programming Language for commercial and systems programming applications. The student is prepared to design, code, test, and execute C Language programs making use of the facilities provided by the language. The course material conforms to the ANSI Standard for C Language.

Objectives:

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

- Discuss the basic structure and syntax of the C Programming Language.
- Describe how an application program interfaces with the operating system.
- Design and code structured C Language programs using all available control structures.
- Compile and execute C Language programs in various operating system environments such as UNIX®, MS-WINDOWS, MS-DOS, and MVS.
- Use the storage management capabilities of the C Programming Language.
- Handle input/output processing using the standard C Language Library routines.
- Effectively use arrays, pointers, and structures within a C Language program.
- Use the C Language bitwise operators.
- Access command line arguments from within a C Language program.

Audience:

Technical Users, Applications Programmers, and Systems Programmers.

Prerequisites:

Working knowledge of another programming language. Knowledge of an operating system and editor for program creation.

Duration:

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

PROGRAMMING IN C LANGUAGE
COURSE OUTLINE

I. **INTRODUCTION**

A. Philosophy Behind C
B. Advantages and Disadvantages
C. Language's Important Features

II. **ORGANIZATION**

A. Overall Organization of C Programs
B. Components of a C Program
C. Simple Function Description
D. Entering and Exiting a C Program

III. **DATA**

A. Basic Data Representation
B. Constants and Variables
C. Identifier Naming Rules

IV. **EXPRESSIONS AND OPERATORS**

A. Identifying and Using Expressions
B. Basic Operators
   1. Assignment
   2. Arithmetic
   3. Compound Assignment
   4. Precedence and Associativity

V. **BASIC I/O**

A. Invoking Simple I/O Functions
B. Terminal Oriented Character I/O
C. Simple Formatted Output

VI. **COMPILATION**

A. Under the UNIX System
B. Under MS/DOS (Microsoft)
C. Under MVS

VII. **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

VIII. **FUNCTIONS I**

A. Traditional and ANSI Formats
B. Passing Arguments by Value

IX. **DATA II**

A. Review Basic Data Types
B. Characteristics of Each Type
C. Use of Constants
D. Data Type Modifiers

X. **BASIC I/O II**

A. `printf` Options
B. `printf` Format Specifications
C. Field Width and Precision

XI. **FUNCTIONS II**

A. Use of `return` Statement
B. Functions Returning Integers
C. Functions Returning Non-Integers
D. Argument Promotion

PanTech
908-281-6759

XII. **ARRAYS AND STRINGS**

A. Definition and Use of Arrays
B. Strings and Character Arrays
C. Arrays as Function Arguments

XIII. **POINTERS**

A. Definition of a Pointer
B. Declaration and Use of Pointers
C. Passing Pointers to Functions
D. Pointer relationship to Array
E. `scanf` Library Routine

XIV. **STRUCTURES**

A. Declaration and Initialization
B. Accessing Structure Members
C. Pointers to Structures
D. Arrays of Structures
E. Structures and Functions
F. Similar Constructs (Union, Enumeration)

XV. **STANDARD I/O**

A. Opening and Closing Files
B. Reading and Writing Files
C. Formatted Output
D. Error Management

XVI. **STORAGE CLASSES**

A. Definition of Storage Class
B. Scope of Identifier Names
C. Storage Class Characteristics
D. Initialization Considerations

XVII. **BIT OPERATORS**

A. AND Operator
B. Inclusive and Exclusive OR Operators
C. Ones Complement Operator
D. Shift Operators

XVIII. **POINTERS II**

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

XIX. **APPENDICES**

A. Lab Exercises
B. Lab Solutions
C. The ANSI Standard
D. C More Clearly - Functions
E. C More Clearly - Bit Operators
F. C More Clearly - Conversions
G. ANSI Extensions for Characters
H. Compiler and Environmental Considerations
I. BUG Alerts
J. Suggested Reading List
K. Preprocessor Overview
L. C More Clearly - Operator Precedence