

Course duration

- 5 days

Course Benefits

- Learn about what C is.
- Work with data types and variables.
- Work with pointers and arrays.
- Learn about structures.
- Learn what they are and how to work with operators and expressions.
- Work with control flow statement.
- Work with functions.

Course Outline

1. Introduction to C
 1. What Is C?
 2. Features of C
 3. Why Program in C?
 4. History of C
 5. Current Status and Future
2. An Overview of C
 1. The First Program (hello.c)
 2. How to Compile and Run a C Program
 3. An Arithmetic Program (roof.c)
 4. Execution Flow Control (mph.c)
 5. The for Loop
 6. The for Loop
 7. Diagram
 8. Character I/O
 9. A File Copier Program (cp2.c)
 10. A Character Counter (wc2.c)
 11. A Look at Arrays
 12. Stock Values (stock1.c)
 13. The char Data Type
 14. Strings (Character Arrays)
 15. A String Copy Program (stringcp.c)
 16. A Look at Functions
 17. A Functional Program (func1.c)
 18. A Review of printf()
3. Data Types and Variables

1. Fundamental Data Types
2. Data Type Values and Sizes
3. Variable Declarations
4. Variable Names
5. Constants
6. Character Constants
7. String Constants
4. Operators and Expressions
 1. What Are Expressions?
 2. Arithmetic Operators
 3. Relational Operators
 4. Assignment Operator
 5. Expressions Have Resulting Values
 6. True and False
 7. Logical Operators
 8. Increment and Decrement Operators (++ and --)
 9. Increment and Decrement Operators: Examples
 10. 'Operate-
 11. Assign' Operators (+=, *=, ...)
 12. Conditional Expression
 13. Operator Precedence
 14. Precedence and Order of Evaluation
 15. Evaluation of Logical Operators
 16. Type Conversions
 17. The Cast Operator
 18. Bitwise Logical Operators
5. Control Flow
 1. Statements
 2. if - else
 3. if() - else if()
 4. else if()
 5. switch()
 6. while()
 7. do-while()
 8. for()
 9. The for Loop-Diagram
 10. Example: for() Loop
 11. Another Example: for() Loop
 12. The break Statement
 13. The continue Statement
6. Functions
 1. What Is a Function?
 2. Example: findbig3()
 3. Why Use Functions?
 4. Anatomy of a Function
 5. Example: find_big_int()
 6. Arguments Passed by Value

7. Addresses of Arguments Can Be Passed
8. A Picture of Addresses and Values
9. When to Use the Return Statement
10. Returning Non-Integer Values
11. Functions in Multiple Source Files
12. A Simple make File
13. The Concept of Variable Scope
14. Automatic Variables
15. Global (External) Variables
16. Static Variables
17. External Static Variables
7. The C Preprocessor
 1. Symbolic Constants
 2. Macro Substitution
 3. File Inclusion
8. Pointers and Arrays
 1. What Is a Pointer?
 2. Pointer Operators
 3. Example: Pointers
 4. Why Use Pointers?
 5. Arrays
 6. Arrays (a Picture)
 7. The & Operator
 8. Pointers and Arrays
 9. Pointer Arithmetic
 10. Pointer Arithmetic (a Picture)
 11. Arrays and Pointers
 12. Array Names are Constant Pointers
 13. Passing A
 14. Arrays to Functions
 15. Initializing Arrays
9. Advanced Pointers
 1. Pointer Initialization
 2. Command-line Arguments
 3. Strings and Character Pointers
 4. Arrays of Pointers
 5. Command-Line Arguments
 6. Access Through Pointers
 7. Functions and Pointers
 8. Example: Functions and Pointers
10. Structures
 1. Structures
 2. Comparison of Structures and Arrays
 3. Structure Definitions
 4. Structure Declarations
 5. Structure Parameter Passing by Reference
 6. Pointers to Structures

7. Structure Parameter Passing Again
8. Arrays of Structures
9. The malloc Routine

Class Materials

Each student will receive a comprehensive set of materials, including course notes and all the class examples.

Class Prerequisites

Experience in the following *is required* for this C/C++ class:

- Programming experience with the desire and need to learn the C language.

Experience in the following *would be useful* for this C/C++ class:

- The ability to program in a high-level language such as a Pascal or COBOL is very helpful.
- Basic UNIX user-level skills are also important.