

C Programming: An Introduction

Course length: 5 days

Course Description

You have used many software applications. Creating a software application involves writing instructions that the computer understands and performing the expected task. The language used to write these instructions is called a programming language. In the C Programming: An Introduction (2008 Update) course, you will learn the basic concepts of writing a program using the C language with a special emphasis on the modular approach to programming.

Course Objective: You will learn the basic programming concepts and develop programs using the modular approach to the C programming language.

Target Student: The target student for this course is an entry-level programmer interested in developing applications using C. This is for the serious programming student and basis for continued education in C, C++, or Java programming.

Prerequisites: Participants should be familiar with general programming concepts. Experience with scripting languages such as Visual Basic or Unix Shell scripting would be helpful. The suggested prerequisite course is Introduction to Programming Using C.

Delivery Method: Instructor led, group-paced, classroom-delivery learning model with structured hands-on activities.

Performance-Based Objectives

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

- explore the basics of the C language.
- execute a simple C program.
- use the basic building blocks of C to write simple programs.
- create arithmetic expressions to perform mathematical calculations.
- use basic input/output functions.
- use control-flow branching statements to selectively execute program code depending on a test condition.
- use control-flow looping statements.
- use arrays to store more than one value in a variable.
- work with strings.
- declare and initialize pointers and use them to access data and perform mathematical operations.
- use functions in a C program.
- work with standard C library functions.
- work with functions and character strings.
- pass data to functions.
- use storage classes.
- define structures in a C program.
- work with unions to create a collection of variables with minimal memory utilization.
- resolve errors in C programs.
- manage dynamic memory allocation.
- work with bitwise operators.
- use preprocessor directives.
- perform multifile compilation.
- use file I/O functions.
- work with data structures.



Course Content

Lesson 1: Getting Started with C

Topic 1A: Explore the C Language
Topic 1B: Identify the Elements of the C Program

Lesson 2: Executing a Simple C Program

Topic 2A: Write a Basic C Program
Topic 2B: Compile the C Program
Topic 2C: Debug the Program
Topic 2D: Run a C Program

Lesson 3: Using the Basic Building Blocks of C

Topic 3A: Identify the Basic Building Blocks of C
Topic 3B: Use Data in a C Program
Topic 3C: Perform Forced Type Casting

Lesson 4: Creating Arithmetic Expressions

Topic 4A: Identify the Basic Arithmetic Operators
Topic 4B: Use Arithmetic Operators to Create an Expression

Lesson 5: Using Input/Output Functions

Topic 5A: Understand I/O Implementation
Topic 5B: Use the getchar() and the putchar() Functions
Topic 5C: Use the gets() and the puts() Functions
Topic 5D: Use Formatted I/O

Lesson 6: Using Control-Flow Branching Statements

Topic 6A: Use Relational Operators
Topic 6B: Use Logical Operators
Topic 6C: Write Conditional Statements

Lesson 7: Using Control-Flow Looping Statements

Topic 7A: Use the while Loop
Topic 7B: Use the for Loop

Lesson 8: Working with Arrays

Topic 8A: Create an Array
Topic 8B: Initialize an Array

Lesson 9: Working with Strings

Topic 9A: Initialize a String
Topic 9B: Perform String Operations

Lesson 10: Working with Pointers

Topic 10A: Declare a Pointer Variable
Topic 10B: Initialize a Pointer
Topic 10C: Assign Pointers to an Array
Topic 10D: Access Values in an Array Using Pointers



Lesson 11: Using Functions in a C Program

Topic 11A: Understand Functions

Topic 11B: Declare a Function

Topic 11C: Write a Function to Return a Value

Lesson 12: Working with Library Functions

Topic 12A: Examine Standard Library Functions

Topic 12B: Create a Library Function

Lesson 13: Working with Functions and Character Strings

Topic 13A: Pass Arrays to Functions

Topic 13B: Use String Conversion Functions

Lesson 14: Passing Data to Functions

Topic 14A: Pass Variables as Arguments

Topic 14B: Change the Variable Value by Passing Its Address

Lesson 15: Using Storage Classes

Topic 15A: Declare Global Variables

Topic 15B: Declare Automatic Variables

Topic 15C: Declare External Variables

Topic 15D: Identify Static Variables

Topic 15E: Create Register Variables

Lesson 16: Defining Structures in a C Program

Topic 16A: Create a Structure

Topic 16B: Initialize a Structure

Topic 16C: Create an Array of Structures

Topic 16D: Pass Structures to Functions

Topic 16E: Use Structure Pointers

Topic 16F: Create Nested Structures

Lesson 17: Working with Unions

Topic 17A: Create a Union

Topic 17B: Access Union Fields

Topic 17C: Examine Bit-Fields and Enumerations

Lesson 18: Eliminating Errors in a C Program

Topic 18A: Resolve Compile-Time Errors

Topic 18B: Resolve Link-Time Errors

Topic 18C: Resolve Runtime Errors

Lesson 19: Managing Dynamic Memory Allocation

Topic 19A: Allocate a Memory Block

Topic 19B: Release Dynamic Memory

Topic 19C: Reallocate Memory

Lesson 20: Working with Bitwise Operators

Topic 20A: Manipulate Data Using Bitwise Operators

Topic 20B: Swap Variables

Topic 20C: Perform Arithmetic Calculations



Lesson 21: Using the Preprocessor Directives

Topic 21A: Define a Preprocessor Directive

Topic 21B: Use Conditional Compilation Directives

Topic 21C: Use Preprocessor Operators

Topic 21D: Understand Predefined Macros

Lesson 22: Performing Multifile Compilation

Topic 22A: Compile Multiple Files

Topic 22B: Pass Data to Functions in Separate Files

Topic 22C: Share Global Data in Files

Lesson 23: Using File I/O Functions

Topic 23A: Use Command-Line Arguments

Topic 23B: Work with Files

Lesson 24: Working with Data Structures

Topic 24A: Work with Stacks

Topic 24B: Work with Queues

Topic 24C: Work with Linked Lists

