Website : www.pineappleeducation.com
PINEAPPLE EDUCATION
Innovation, Research, Academic & Training Institute with Placement Support
(Mentored by Lawyers, Chartered Accountants, IT Professionals & Academicians)
21 Years
Powered by IIMMI INTERNATIONAL INSTITUTE OF MANAGEMENT, MEDIA & IT

PROFESSIONAL CORE C PROGRAMMING PROGRAM | CAREER-FOCUSED CURRICULUM

Learning Roadmap

A visual path through the complete 7-module C Programming curriculum.

2 Months	7 Modules	Core C + Pointers + Projects
Program Duration	Complete Curriculum	Applied Learning

01 Programming Fundamentals & C Setup
Foundation

02 Control Flow & Functions
Logic & Reusability

03 Arrays & Strings
Working with Data

04 Pointers & Memory Management
Pointers & Memory

05 Structures, Unions & File Handling
Persistence & Records

06 Advanced C & Data Structures
Advanced C & Tools

07 Real-World Projects
Portfolio on GitHub

Program at a Glance

Key information about the Core C Programming program.

Program	Professional Core C Programming Program
Duration	2 Months (approx. 8 weeks), 7 modules
Learning Approach	Instructor-led sessions, hands-on labs, module assignments and portfolio projects
Eligibility	Graduates and final-year students from any stream; basic computer knowledge. No prior programming experience required
Technology Stack	C (C11 standard), GCC, VS Code, GDB, Git and GitHub, Make, Valgrind
Program Outcome	Confident C programmer with a strong grip on pointers and memory, who can write clean, modular programs and build complete applications

Program Structure & Timeline

Module-wise duration, phase and primary tools, progressing from C fundamentals to pointers, data structures and projects.

No.	Module	Phase	Duration	Key Tools
01	**Programming Fundamentals & C Setup**	Foundation	1 Week	GCC, VS Code, Git, Terminal
02	**Control Flow & Functions**	Core C	1 Week	GCC, VS Code, GDB Debugger
03	**Arrays & Strings**	Core C	1 Week	Arrays, Strings, string.h
04	**Pointers & Memory Management**	Core C	1 Week	Pointers, malloc, calloc, free
05	**Structures, Unions & File Handling**	Core C	1 Week	struct, typedef, fopen, fread, fwrite
06	**Advanced C & Data Structures**	Advanced	1 Week	GCC, GDB, make, Valgrind
07	**Real-World Projects**	Projects	2 Weeks	C, Pointers, Structures, File Handling, Git
Total Program Duration			**8 Weeks**	Approx. 2 months

MODULE 01

Programming Fundamentals & C Setup

01

Foundation-level introduction to programming, C installation and the core building blocks: variables, data types, operators and input/output.

DURATION	LEVEL	KEY TOOLS	MODULE OUTCOME
1 Week	Beginner	GCC, VS Code, Git, Terminal	Working C environment and the ability to write, compile and run basic programs

Part 1: Getting Started

- Introduction to Programming
- Structure of a C Program
- Command Line Basics
- History and Features of C
- Compilation Process
- Version Control Basics (Git)
- Installing GCC and VS Code
- Comments and Indentation

Part 2: Variables & Data Types

- Variables and Naming Rules
- Type Modifiers
- Format Specifiers
- int, float, double, char
- Type Casting
- Constants and #define
- printf() and scanf()

Part 3: Operators

- Arithmetic Operators
- Assignment Operators
- Ternary and sizeof
- Relational Operators
- Bitwise Operators
- Operator Precedence
- Logical Operators
- Increment and Decrement

Software to Install

- GCC / MinGW
- Terminal / Command Prompt
- Visual Studio Code
- Git and GitHub

Practice Programs

- **Simple Calculator**
- **Temperature Converter**
- **Area and Perimeter Calculator**
- **Swap Two Numbers**

WHAT THIS MODULE SUPPORTS

SETUP
Configure GCC, VS Code and Git on your machine.

WRITE
Create, compile and run your first C programs.

CALCULATE
Work with data types, operators and user input.

MODULE 02

Control Flow & Functions

02

Decision making, repetition and code reuse: conditional statements, loops, functions, storage classes, recursion and the preprocessor.

DURATION	LEVEL	KEY TOOLS	MODULE OUTCOME
1 Week	Beginner	GCC, VS Code, GDB Debugger	Write logical, reusable programs using conditions, loops and functions

Part 1: Conditional Statements

- if and if-else
- else-if Ladder
- Nested Conditions
- switch-case Statement
- Ternary Operator

Part 2: Loops

- for Loop
- while Loop
- do-while Loop
- break and continue
- goto (Awareness)
- Nested Loops
- Pattern Printing

Part 3: Functions

- Declaration, Definition, Call
- Parameters and Arguments
- Return Values
- Call by Value
- Scope and Storage Classes
- auto, static, extern, register
- Recursion
- Header Files

Part 4: Preprocessor

- #include and #define
- Macros with Arguments
- math.h and stdlib.h
- Creating Your Own Header

Practice Programs

- **Number Guessing Game**
- **Prime and Fibonacci Series**
- **Multiplication Table**
- **Menu-driven ATM Program**

WHAT THIS MODULE SUPPORTS

DECIDE
Control program flow with conditions.

REPEAT
Automate repetition with loops.

REUSE
Organize code into functions and headers.

MODULE 03

Arrays & Strings

03

Working with collections of data: one and two-dimensional arrays, searching and sorting, character arrays and the string library.

DURATION	LEVEL	KEY TOOLS	MODULE OUTCOME
1 Week	Beginner to Intermediate	Arrays, Strings, string.h	Store, organize and process collections of data using arrays and strings

Part 1: Arrays

- One-Dimensional Arrays
- Two-Dimensional Arrays
- Array Initialization
- Matrix Operations
- Traversal and Updates
- Passing Arrays to Functions

Part 2: Searching & Sorting

- Linear Search
- Selection Sort
- Binary Search
- Insertion Sort
- Bubble Sort
- Finding Min, Max, Sum

Part 3: Strings

- Character Arrays
- strlen, strcpy, strcat
- Array of Strings
- Null Terminator
- strcmp and strstr
- fgets() and puts()
- String Manipulation

Practice Programs

Matrix Add and Multiply

Student Marks Record

Word Counter

Palindrome and Anagram Checker

WHAT THIS MODULE SUPPORTS

STORE
Choose the right layout for your data.

PROCESS
Search, sort and transform collections.

MANIPULATE
Work confidently with text and strings.

MODULE 04

Pointers & Memory Management

04

The heart of C: pointers, address arithmetic, dynamic memory allocation and safe memory handling.

DURATION	LEVEL	KEY TOOLS	MODULE OUTCOME
1 Week	Intermediate	Pointers, malloc, calloc, free	Use pointers confidently and manage memory dynamically without leaks

Part 1: Pointer Basics

- Address-of and Dereference
- Pointer Arithmetic
- Declaring Pointers
- Pointers and Arrays
- NULL and void Pointers
- Pointers and Strings

Part 2: Advanced Pointers

- Pointer to Pointer
- Function Pointers
- Call by Reference
- Dangling and Wild Pointers
- Returning Pointers
- const with Pointers

Part 3: Dynamic Memory

- Stack vs Heap
- Dynamic 1D and 2D Arrays
- malloc() and calloc()
- Memory Leaks
- realloc() and free()
- Common Pointer Bugs

Practice Programs

Swap Using Pointers

Dynamic Array Resizer

Custom strlen and strcpy

Dynamic Student List

WHAT THIS MODULE SUPPORTS

ADDRESS
Understand how memory really works.

ALLOCATE
Create data at runtime with the heap.

DEBUG
Avoid leaks and pointer errors.

MODULE 05

Structures, Unions & File Handling

05

Modelling real-world data with structures and persisting it to disk with text and binary files.

DURATION	LEVEL	KEY TOOLS	MODULE OUTCOME
1 Week	Intermediate	struct, typedef, fopen, fread, fwrite	Design record-based programs and read, write and update files reliably

Part 1: Structures & Unions

- Defining struct
- Array of Structures
- Nested Structures
- Pointers to Structures (->)
- typedef
- union and enum
- Bit-fields Basics

Part 2: File Handling

- fopen() Modes and fclose()
- fprintf() and fscanf()
- fgets() and fputs()
- Binary Files: fread, fwrite
- fseek, ftell, rewind
- Updating and Deleting Records

Part 3: Robust Programs

- Error Handling with perror
- Command-line Arguments
- Multi-file Programs
- Input Validation
- Debugging Techniques

Practice Programs

Student Record System

Bank Account Manager

Employee Database (Binary)

Log File Reader

WHAT THIS MODULE SUPPORTS

MODEL
Represent entities with structures.

PERSIST
Save and load data from files.

PROTECT
Handle errors without crashing.

MODULE 06

Advanced C & Data Structures

06

Dynamic data structures built with pointers, multi-file project organisation and professional debugging tools.

DURATION	LEVEL	KEY TOOLS	MODULE OUTCOME
1 Week	Intermediate	GCC, GDB, make, Valgrind	Build linked structures and ship clean, debugged, multi-file C projects

Part 1: Data Structures

- Singly Linked List
- Doubly Linked List (Intro)
- Stack using Array and List
- Queue using Array and List
- Applications of Stack
- Complexity Basics

Part 2: Advanced Language Features

- Bitwise Manipulation Tricks
- Header Guards
- Conditional Compilation
- Variadic Functions
- Function Pointers and Callbacks
- Static and Shared Libraries

Part 3: Tools & Code Quality

- Debugging with GDB
- Memory Checks with Valgrind
- Makefile Basics
- Coding Standards
- Writing Clean, Commented Code

Practice Programs

- **Linked List Manager**
- **Parentheses Checker (Stack)**
- **Bit Manipulation Utilities**
- **Multi-file Math Library**

WHAT THIS MODULE SUPPORTS

BUILD
Implement core data structures from scratch.

DEBUG
Find bugs and leaks with real tools.

SHIP
Organize and document clean C projects.

MODULE 07

Real-World Projects

07

Applied portfolio work designed to reinforce C fundamentals, pointers, structures, file handling and problem-solving skills through complete working programs.

DURATION	LEVEL	KEY TOOLS	MODULE OUTCOME
2 Weeks	Applied Practice	C, Pointers, Structures, File Handling, Git	Complete a portfolio of working C projects on GitHub

Project List

- 01 Calculator App
- 02 Number Guessing Game
- 03 To-Do List Manager
- 04 Contact Book
- 05 Expense Tracker
- 06 Library Management System
- 07 Student Management System
- 08 ATM Banking System

Portfolio Practice

Build console-based projects using the concepts from the previous modules, progressing from small programs to complete menu-driven applications with structures, dynamic memory, file storage and error handling. Each project emphasizes planning, clean code, working features and a well-documented GitHub repository.

This curriculum is suitable for a 2-month professional training program, progressing from C fundamentals to pointers, structures, file handling and real-world applications.