

Advanced C Programming

Audience Course Advanced C Programming

The course Advanced C Programming is intended for C developers who want to understand and use the more advanced features and techniques of the C language.

Prerequisites Course Advanced C Programming

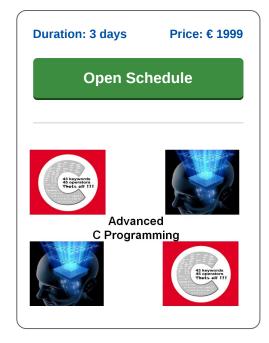
To join this course knowledge of the basic concepts of $\underline{\text{programming in C}}$ and experience in C is required.

Realization Training Advanced C Programming

The concepts are treated on the basis of presentation slides. Illustrative demo programs are used to clarify the treated concepts. There is ample opportunity to practice and theory and practice is interspersed.

Official Certificate Advanced C Programming

Participants receive an official certificate Advanced C Programming after successful completion of the course.



Content Course Advanced C Programming

In the course Advanced C Programming the more advanced aspects of C programming like the use of modules and interfaces and object oriented programming in C using encapsulation are discussed.

Potential Dangers in C

Potential dangers of certain constructs in C are highlighted such as C macros, evaluation order and the use of globals.

Pointer Techniques

Attention is paid to pointer techniques and the use of pointers to functions is discussed.

Dynamic Data Structures

The most common dynamic data structures such as arrays, linked lists, hash tables and trees and their implementation in C are addressed in detail.

Optimization Techniques

Also attention is paid to optimization techniques regarding speed and memory.

Advanced C Library Function

More advanced functions of the C library such as setjmp, longjmp, signals, bsearch, and qsort etc. are also on the course program.

Bit Manipulation

Finally bit manipulation and advanced string handling and parsing are discussed.

Houten, Amsterdam, Rotterdam, Eindhoven, Zwolle, Online



Modules Course Advanced C Programming

Module 1 : C Pitfalls	Module 2 : Object Orientation in C	Module 3 : Pointers and Arrays
C Standards	Object Orientation	Pointers Revisited
Pitfalls C Language	Classes and Objects	Pointer Arithmetic
Write Clean Code	Example Class and Objects	Pointers and Arrays
Good Programming Style	Object Orientation in C	Function Pointers
Rules for Functions	Simple Objects in C	Optional Ampersand
Deep versus Flat Code	C versus C++	Variable Length Arguments
Evaluation Order	Class Constructor in C	Varargs Example
Avoid Macros	Member Functions in C	Array Initialization
Without const	Encapsulation in C	Array Traversal
Using const	Public Function Members in C	Arrays of Structs
Code Reuse	Inheritance	Arrays versus Pointers
Avoid Globals	Extending C++ Classes	Multidimensional Arrays
Pre and Post Conditions	Virtual Functions	Indices in 3d Array
Invariants	Virtual Function Table	Dynamic Arrays
Code Smells	Polymorphism	Ragged Arrays
Module 4 : Data Structures in C	Module 5 : Bit Manipulation	Module 6 : C Standard Library
Dynamic Data Structures	Bit Manipulation	What are Signals?
Singly Linked Lists	Bitwise Operators	ANSI C-Signal Types
Linked List Element Type	Bitwise AND	Handling Signals
Creating List Elements	Bitwise OR	SIGABRT, SEGV, SEGFPE
Doubly Linked Lists	Bitwise XOR	setimp and longimp
Stacks and Queues	Bitwise NOT	Coroutines
Hash Tables	Bitshift Operators	atexit Function
Load Factor	Bit Shifting	assert Function
Hash Functions	Bitwise Assignment Operators	perror Function
Trees	Bit Rotation	Raising Signals
Trees Traversal	BitFlag Functions	Alarm Signal
Module 7 : String Handling		-
Looking for Characters		
	I .	
Looking for Substrings		
Looking for Substrings Count Matching Characters		
-		

String Comparison String Tokenizing

qsort and Bsearch Advanced String Handling

Converting Strings to Numbers Handling Conversion Errors