

Price: R6,900.00 excl. VAT
Duration: 5 days
Code: SANSC

Standard C Programming

Description

This is a high content course that provides a complete overview of the C language, based on the ANSI standard. The focus is on programming standard C in a structured and portable manner, and reliance is not placed on any specific compiler or platform.

Objectives

Delegates who complete this course will be able to:

- Understand the C compiler and the pre-processor.
- Use pre-processor directives.
- Develop portable, standards compliant code using C language constructs.
- Read, debug and maintain C code.
- Understand the importance of portable code and standards compliance.

Intended Audience

Any programmer who needs to develop or maintain systems written in C. Developers who have had limited exposure to C, or are self-taught, will benefit from this course.

Prerequisites

Previous programming experience is essential. Persons with some C background or who are self-taught in C will derive great benefit from the course.

Course Contents

Introduction • Overview of the C language. • Portability. • Terminology. • Programming fundamentals. • Structured programming principles.

The C Compiler • Compiler operation in general. • The preprocessor. • Output results of the compilation phases. • Object files and libraries. • Installing a C compiler. • Creating program files. • Compiler code generation, the linking process and executing programs. • The C startup module and the main function. • Separate compilation.

C Language Fundamentals • Keywords. • Program structure and conventions. • The standard library. • Header files. • Functions and I/O functions. • Character set. • Literal values. • Comments. • Preprocessor directives. • Fundamental types, derived types, structured types, enumerated types, user defined types. • Storage classes.

Expressions and Operators • Expressions and expression results. • Operators in expressions. • Type requirements of operators. • Implicit and inherent type conversions. • Types of operands and the resulting types. • Bitwise, relational, logical, compound assignment and other operators.

Functions and Statements • Function definitions, declarations and prototypes. • Passing parameters. • Recursive functions. • Function returns. • The function call operator. • Local variables. • Function pointers and the pointers-to-function data type. • Types of statements.

Pointers, Arrays and Structs • Pointer types and operators. • Pointer arithmetic. • Array subscripts. • Indirection. • Multi-dimensional arrays. • Pointer-to-array and pointer-to-function types. • Structs and unions. • Member selection, indirect member selection.