

PRG300: Cursus C++ Programmeren

Code: PRG300

Duur: 5 dagen

Prijs: € 2200

Doelgroep C++ Opleiding

De cursus C++ Programmeren is bedoeld voor developers die in C++ willen leren programmeren en anderen die C++ code willen begrijpen.

Voorkennis voor de Cursus C++

Om aan deze cursus te kunnen deelnemen is kennis van en ervaring met programmeren in C vereist.

Uitvoering Training Cursus C++

De theorie wordt behandeld aan de hand van presentatie slides en wordt afgewisseld met oefeningen. Illustratieve demo's worden gebruikt om de behandelde begrippen te verduidelijken. De cursustijden zijn van 9.30 tot 16.30.

Certificering C++

De deelnemers krijgen na het goed doorlopen van de training een officieel certificaat C++ Programmeren.

Inhoud van de Training C++

In de cursus C++ Programmeren leren de deelnemers programmeren in de C++ programmeertaal. In de cursus wordt de nieuwste C++ 17 standaard gebruikt. Eerst worden de verschillen tussen C en C++ besproken voor wat betreft declaraties van variabelen, formatted output met de stream IO library, namespaces, function overloading en default function parameters. Vervolgens wordt aandacht besteed aan de nieuwe C++ reference variabelen. Zowel Lvalue als Rvalue references worden besproken. Een belangrijk onderdeel van de cursus is het C++ class concept en de C++ implementatie van object georiënteerde principes als abstraction en encapsulation. Ook wordt aandacht besteed aan dynamische geheugen allocatie met new en delete en de rol van assignment operators en copy en move constructors. Ook speciale kenmerken van classes, zoals statics, friends en iterators worden besproken. Vervolgens staan ook de object georiënteerde principes van inheritance en polymorfisme op het programma. Daarbij komen de concepten van virtual functions, v-tables, dynamic binding en abstract classes ter sprake. In C++ is het mogelijk om standaard operatoren een andere betekenis te geven en dit fenomeen wordt besproken in de module operator overloading. Vervolgens komen belangrijke kenmerken van de C++ standard library aan bod zoals de String class en de principes van C++ templates en de Standard Template Library (STL). Tot slot wordt aandacht besteed aan exception handling en hoe dit is geïmplementeerd in C++.



C++ Programming



Module 1 : Intro C++	Module 2 : Variables and Types	Module 3 : References
Intro C++ C++ TimeLine Comments in C++ Namespace std Output and Error Stream Standard Input Stream cin and Strings Formatted Output Variable Declaration Scope Resolution Operator Inline Functions Default Function Arguments Overloading Functions Range based for loop	Standard Types Type Inference Auto Keyword Deduction with decltype Initialization Null Pointer Constant Strongly Types Enums Variable Scope Namespaces Using keyword and Directive Block Usage User Defined Literals Storage Classes const Qualifier	References Reference Initialization References and Pointers Rvalues and Rvalues in C Rvalues and Rvalues in C++ Reference to Constant Passing References Comparison Parameter Passing References as Return Values Returning lvalue Returning Reference to Global Rvalue References Comparing Reference Types Rvalue Reference Usage
Module 4 : Classes	Module 5 : Dynamic Memory Allocation	Module 6 : Inheritance
Classes and Objects Classes in C++ Class Declaration Class Sections Constructor and Destructor Uniform Initialization Header and Sources Files Class Implementation Advantages Access Functions References to private Data this Pointer static Members Constant Objects Member Objects Friends	new and delete Operators Dynamic Arrays Classes with Pointer Data Assignment Operator Self-Assignment Problem Chained Assignments Assignment and Initialization Copy Constructors Passing Objects Returning Objects Passing References to Objects Move Constructor Move Assignment Operator Perfect Forwarding Delegating Constructors	Inheritance Derived Classes in C++ Class Hierarchy Redefining Member Functions Derived Class Constructors Base – Derived Class Conversion Pointer Conversions Virtual Functions Polymorphism Dynamic Binding Virtual Function Table Pure Virtual Functions Abstract Classes Multiple Inheritance Virtual Derivation

Module 7 : Operator Overloading	Module 8 : Exception Handling	Module 9 : Templates
<ul style="list-style-type: none"> Operator Overloading Overloading for Numeric Types Complex Type Example Overloading Rules Overloading Restrictions Not Overloadable Operators When not to Overload Numeric Class Overloading Operators as Friend Unary Overloading Operator 	<ul style="list-style-type: none"> Exception Handling in C++ Memory Exhaustion Handling Throwing Exceptions try Block catch Handlers Multiple catch Handlers Template Array Class Exceptions Array Class catch Order throw List 	<ul style="list-style-type: none"> What are Templates? Template Functions Template Specialization Template Parameter List Class Templates Template Parameter Scope Template Function Statics Template Class Statics Inclusion Compilation Model Templates and Friends
<p>Module 10 : STL</p>		
<ul style="list-style-type: none"> Standard Template Library STL Core Components STL Library Components STL Containers Vector Container Deque Container List Container STL Iterators STL Algorithms STL Allocators 		