C++ 11-14-17-20

Doelgroep Cursus C++ 11-14-17-20
De cursus C++ 11-14-17-20 is bedoeld voor C++ developers die de nieuwe features van de C++ 11, 14, 17 en 20 standaarden willen leren kennen.

Voorkennis Cursus C++ 11-14-17-20
Om aan deze cursus te kunnen deelnemen is kennis van en ervaring met programmeren in C++ vereist.

Uitvoering Training C++ 11-14-17-20
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 cursus tijden zijn van 9.30 tot 16.30.

Certificering cursus C++ 11-14-17-20
De deelnemers krijgen na het goed doorlopen van de training een officieel certificaat C++ 11-14-17-20.

Duur: 2 dagen **Prijs: € 1499**

Open Rooster

C++ 11-14-17-20

Inhoud Cursus C++ 11-14-17-20

In de cursus C++11-14-17-20 leren de deelnemers de nieuwe features die aan de C++ standaard zijn toegevoegd sinds C++11. De taal C++ is niet weg te denken uit de wereld van high performance en embedded software.

C++ Standaard
Sinds de C++ 98 standaard bleef het lange tijd stil, maar sinds C++11 maakt C++ de laatste jaren een gestage ontwikkeling door met de releases van C++14, C++17 en C++20.

C++11 Features
De cursus gaat van start met een overzicht van een aantal nieuwe features in C++11 zoals type inference, uniform initialization, lambda functions en strongly typed enums.

Rvalue Reference
Vervolgens wordt ingegaan op de nieuwe Rvalue reference types in C++11 waardoor move constructors in plaats van copy constructors mogelijk worden. Ook wordt daarbij de Golden Rule of 5 voor classes met pointer members besproken.

Smart Pointers
Een belangrijke toevoeging aan de C++ standaard die met C++11 is geïntroduceerd zijn de smart pointers. Deze zijn van belang bij het voorkomen van memory leaks en zowel unique, shared en weak smart pointers komen aan de orde.

Multiple Threads
Verder is er aandacht voor multiple threads die sinds C++11 onderdeel zijn van de C++ standaard. Met threads kunnen C++ applicaties geparallelliseerd worden en optimaal gebruik maken van multiple cores. Ook de synchronisatie van threads door locking mechanismen als mutexes en condition variables zijn onderdeel van het cursus programma.

C++14 Features
De nieuwe features van C++14 waaronder generalized lambda captures, return type deduction, shared mutexes en shared locking komen vervolgens aan bod.

C++17 en C++20 Features
Tenslotte is er aandacht voor wijzigingen en toevoegingen aan de standaard die gekomen zijn met de release van C++17 en C++20 zoals folding expressions, optional types en immediate functions.

Modules Cursus C++ 11-14-17-20

Module 1 : C++ 11 Intro	Module 2 : Move Semantics	Module 3 : Smart Pointers
C++11 Features	Lvalues and Rvalues in C++	unique_ptr
Type Inference	Reference to Constant	Using unique_ptr
Auto and decltype	Passing References	Specialization for Arrays
Uniform Initialization	References as Return Values	Replacement for std::auto_ptr
Initializer Lists	Rvalue References	std::make_unique
Range Based for Loop	Rvalue Reference Usage	shared _ptr Pointer
Null Pointer Constant	Assignment Operator	Control Block
Standard Types	Copy Constructor	shared_ptr Destruction Policy
constexpr Keyword	Passing and Returning Objects	shared_ptr Interface
Static Asserts	Passing References to Objects	Cyclic References
Lambda Functions	Move Constructor	std::enable_shared_from_this
Strongly Types Enums	Move Semantics	weak_ptr
User Defined Literals	Move Assignment Operator	default Keyword
Raw String Literals	Golden Rule of 5	delete Keyword
Module 4 : Multiple Threads	**Module 5 : Synchronization**	**Module 6 : C++ 14**
Multiple Threads	Data Corruption	Binary Literals
Benefits and Drawbacks	Lock Guard	Generic Lambda Expressions
Thread Characteristics	Automatic Lock Management	Generalized Lambda Captures
Thread Class	Mutex and RAII	Lambda Capture Initializers
Simple Threads	Recursive Locking	Return type Deduction
Joining Threads	Timed Locking	decltype(auto)
Detaching Threads	Atomic Types	Constraints constexpr Functions
Thread ID	Call Once	Variable Templates
Callables	Event Handling	[[deprecated]] Attribute
Passing Parameters	Condition Variables	Digit Separators
Pass by Reference	Wait and Notify	Shared Mutexes
Pass by std::ref and std::move	Promises and Futures	Shared Locking
Member Function as Thread	Asynchronous Tasks	Tuple Addressing via Type
Thread Local Storage	Working with async	std::is_final
Module 7 : C++ 17	**Module 8 : C++ 20**	
Folding Expressions	Concepts Library	
constexpr Lambda	Designated Initializers	
const if Expressions	likely and unlikely Attributes	
Init-statement for if	Immediate Functions	
Inline Variables	constexpr Virtual Functions	
Nested Namespaces	Explicit Bool	
Structured Bindings	Template Syntax Lambdas	
Selection Initialization	Math Constants	
UTF-8 Character Literals	Synchronized Outputstreams	
Initialization of Enums	std::is_constant_evaluated	
fallthrough and nodiscard	std::span	
Guaranteed Copy Elision	std::bit_cast	
Optional Types	std::midpoint	

SpiralTrain BV
Standerdmolen 10, 2e verdieping
3995 AA Houten

info@spiraltrain.nl
www.spiraltrain.nl
Tel.: +31 (0) 30 – 737 0661

Locaties
Houten, Amsterdam, Rotterdam, Eindhoven, Zwolle, Online