

Table of Contents

C++	3
Introduction	3
file Development	3
Debugging & Error Handling	4
Fundamental Data Types & Constants	4
Operators & Bit Manipulation	4
Scope, Duration, and Linkage	4
Control Flow	4
Type Conversion & Deduction	4
Pointers, References & Dynamic Memory	4
Enums and Structs	5
Arrays, Vectors, and Algorithms	5
Advanced Functions & Templates	5
Object-Oriented Programming: Classes	5
Operator Overloading	5
Object Relationships	5
Inheritance & Virtual Functions	5
Advanced Templates	6
Exceptions	6
Move Semantics & Smart Pointers	6
Input and Output (I/O) Streams	6

C++

Snippet from [Wikipedia: C++](#)

C++ is a high-level, general-purpose programming language created by Danish computer scientist Bjarne Stroustrup. First released in 1985 as an extension of the C programming language, adding object-oriented (OOP) features, it has since expanded significantly over time adding more OOP and other features; as of 1997/C++98 standardization, C++ has added functional features, in addition to facilities for low-level memory manipulation for systems like microcomputers or to make operating systems like Linux or Windows, and even later came features like generic programming (through the use of templates). C++ is usually implemented as a compiled language, and many vendors provide C++ compilers, including the Free Software Foundation, LLVM, Microsoft, Intel, Embarcadero, Oracle, and IBM.

C++ was designed with systems programming and embedded, resource-constrained software and large systems in mind, with performance, efficiency, and flexibility of use as its design highlights. C++ has also been found useful in many other contexts, with key strengths being software infrastructure and resource-constrained applications, including desktop applications, video games, servers (e.g., e-commerce, web search, or databases), and performance-critical applications (e.g., telephone switches or space probes).

C++ is standardized by the International Organization for Standardization (ISO), with the latest standard version ratified and published by ISO in October 2024 as *ISO/IEC 14882:2024* (informally known as C++23). The C++ programming language was initially standardized in 1998 as *ISO/IEC 14882:1998*, which was then amended by the C++03, C++11, C++14, C++17, and C++20 standards. The current C++23 standard supersedes these with new features and an enlarged standard library. Before the initial standardization in 1998, C++ was developed by Stroustrup at Bell Labs since 1979 as an extension of the C language; he wanted an efficient and flexible language similar to C that also provided high-level features for program organization. Since 2012, C++ has been on a three-year release schedule with C++26 as the next planned standard.

[Creative Commons Attribution-Share Alike 4.0](#)

Introduction

- [The Compiler & Linker](#)
- [Development Environment \(IDE\)](#)
- [First Program](#)
- [C++ Basics](#)

file Development

- [Function Basics](#)
- [Forward Declarations](#)
- [Multi-file Programs](#)

- [The Preprocessor](#)
- [Namespaces](#)

Debugging & Error Handling

- [Debugging Tactics](#)
- [Using the Debugger](#)
- [Error Detection](#)
- [Code Coverage](#)

Fundamental Data Types & Constants

- [Fundamental Types](#)
- [Signed vs Unsigned](#)
- [Constants & Literals](#)
- [Modern Strings](#)

Operators & Bit Manipulation

- [Operators](#)
- [Precedence & Associativity](#)
- [Bit Manipulation \(Optional\)](#)
- [Bitwise Operators](#)

Scope, Duration, and Linkage

- [Scope & Blocks](#)
- [Linkage](#)
- [Namespaces in Depth](#)
- [Inline Functions](#)

Control Flow

- [Conditionals](#)
- [Switches](#)
- [Loops](#)
- [Random Numbers](#)

Type Conversion & Deduction

- [Implicit & Explicit Casts](#)
- [Type Aliases](#)
- [Type Deduction](#)

Pointers, References & Dynamic Memory

- [L-values & R-values](#)
- [References](#)
- [Pointers](#)
- [Dynamic Allocation \(Legacy\)](#)

Enums and Structs

- Enumerations
- Structs
- Class Templates & CTAD

Arrays, Vectors, and Algorithms

- Dynamic Arrays (std::vector)
- Fixed-size Arrays (std::array)
- Legacy Arrays
- Iterators & Algorithms

Advanced Functions & Templates

- Function Overloading
- Function Templates
- Advanced Pointers
- Lambdas

Object-Oriented Programming: Classes

- Class Basics
- Constructors
- The hidden 'this' pointer
- Destructors
- Friends & Statics

Operator Overloading

- Overloading Methods
- Common Operators
- Advanced Overloading
- Copying

Object Relationships

- Composition
- Aggregation
- Association & Dependencies
- Container Classes

Inheritance & Virtual Functions

- Inheritance
- Multiple Inheritance
- Polymorphism
- Interfaces
- Dynamic Casting

Advanced Templates

- [Class Templates](#)
- [Non-type Parameters](#)
- [Template Specialization](#)

Exceptions

- [Try, Catch, Throw](#)
- [Exception Classes](#)
- [Exception Specifications](#)

Move Semantics & Smart Pointers

- [R-value References \(&&\)](#)
- [Move Semantics](#)
- [Smart Pointers](#)

Input and Output (I/O) Streams

- [Streams](#)
- [String Streams](#)
- [File I/O](#)
- [State Management](#)

The information in this document is cited from [UCH Wiki](#).

From:
<https://wiki.ulascemh.com/> - **UCH**

Permanent link:
<https://wiki.ulascemh.com/doku.php?id=en:cs:lang:c++:start>

Last update: **2026/04/02 16:14**

