C++ is a general-purpose programming language created by Bjarne Stroustrup as an extension of the C programming language. The language has expanded significantly over time, and modern C++ now has object-oriented, generic, and functional features in addition to facilities for low-level memory manipulation. It is almost always implemented as a compiled language, and many vendors provide C++ compilers, including the Free Software Foundation, LLVM, Microsoft, Intel, Oracle, and IBM, so it is available on many platforms.

C++ was designed with a bias toward system programming and embedded, resource-constrained software and large systems, 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 SQL servers), 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 December 2020 as ISO/IEC 14882:2020 (informally known as C++20). 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, and C++17 standards. The current C++20 standard supersedes these with new features and an enlarged standard library. Before the initial standardization in 1998, C++ was developed by Danish computer scientist Bjarne 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++ is on a three-year release schedule, with C++23 the next planned standard.

In 1979, Bjarne Stroustrup, a Danish computer scientist, began work on "C with Classes", the predecessor to C++. The motivation for creating a new language originated from Stroustrup's experience in programming for his PhD thesis. Stroustrup found that Simula had features that were very helpful for large software development, but the language was too slow for practical use, while BCPL was fast but too low-level to be suitable for large software development. When Stroustrup started working in AT&T Bell Labs, he had the problem of analyzing the UNIX kernel with respect to distributed computing. Remembering his Ph.D. experience, Stroustrup set out to enhance the C language with Simula-like features. C was chosen because it was general-purpose, fast, portable and widely used. As well as C and Simula's influences, other languages also influenced this new language, including ALGOL 68, Ada, CLU and ML. Initially, Stroustrup's "C with Classes" added features to the C compiler, Cpre, including classes, derived classes, strong typing, inlining and default arguments. In 1982, Stroustrup started to develop a successor to C with Classes, which he named "C++" (++ being the increment operator in C) after going through several other names. New features were added, including virtual functions, function name and operator overloading, references, constants, type-safe free-store memory allocation (new/delete), improved type checking, and BCPL style single-line comments with two forward slashes (//). Furthermore, Stroustrup developed a new, standalone compiler for C++, Cfront. In 1984, Stroustrup implemented the first stream input/output library. The idea of providing an output operator rather than a named output function was suggested by Doug McIlroy. In 1985, the first edition of The C++ Programming Language was released, which became the definitive reference for the language, as there was not yet an official standard. The first commercial implementation of C++ was released in October of the same year.

In 1989, C++ 2.0 was released, followed by the updated second edition of The C++ Programming Language in 1991.[20] New features in 2.0 included multiple inheritance, abstract classes, static member functions, const member functions, and protected members. In 1990, The Annotated C++ Reference Manual was published. This work became the basis for the future standard. Later feature additions included templates, exceptions, namespaces, new casts, and a Boolean type. In 1998, C++98 was released, standardizing the language, and a minor update (C++03) was released in 2003. After C++98, C++ evolved relatively slowly until, in 2011, the C++11 standard was released, adding numerous new features, enlarging the standard library further, and providing more facilities to C++ programmers. After a minor C++14 update released in December 2014, various new additions were introduced in C++17. After becoming finalized in February 2020, a draft of the C++20 standard was approved on 4 September 2020 and officially published on 15 December 2020. On January 3, 2018, Stroustrup was announced as the 2018 winner of the Charles Stark Draper Prize for Engineering, "for conceptualizing and developing the C++ programming language".

Throughout C++'s life, its development and evolution has been guided by a set of principles:

* It must be driven by actual problems and its features should be immediately useful in real world programs.
* Every feature should be implementable (with a reasonably obvious way to do so).
* Programmers should be free to pick their own programming style, and that style should be fully supported by C++.
* Allowing a useful feature is more important than preventing every possible misuse of C++.
* It should provide facilities for organising programs into separate, well-defined parts, and provide facilities for combining separately developed parts.
* No implicit violations of the type system (but allow explicit violations; that is, those explicitly requested by the programmer).
* User-created types need to have the same support and performance as built-in types.
* Unused features should not negatively impact created executables (e.g. in lower performance).
* There should be no language beneath C++ (except assembly language).
* C++ should work alongside other existing programming languages, rather than fostering its own separate and incompatible programming environment.
* If the programmer's intent is unknown, allow the programmer to specify it by providing manual control.

Benefits of C

* As a middle level language, C combines the features of both high level and low level languages. It can be used for low-level programming, such as scripting for drivers and kernels and it also supports functions of high level programming languages, such as scripting for software applications etc.
* C is a structured programming language which allows a complex program to be broken into simpler programs called functions. It also allows free movement of data across these functions.
* C language is case-sensitive.
* C is highly portable and is used for scripting system applications which form a major part of Windows, UNIX and Linux operating system.
* C is a general purpose programming language and can efficiently work on enterprise applications, games, graphics, and applications requiring calculations.
* C language has a rich library which provides a number of built-in functions. It also offers dynamic memory allocation.