**[cppreference.com](https://en.cppreference.com/)**

Top of Form



Bottom of Form

[Create account](https://en.cppreference.com/mwiki/index.php?title=Special:UserLogin&returnto=cpp%2Flanguage%2Foperator+precedence&type=signup)

* [Page](https://en.cppreference.com/w/cpp/language/operator_precedence)
* [Discussion](https://en.cppreference.com/w/Talk:cpp/language/operator_precedence)
* [View](https://en.cppreference.com/w/cpp/language/operator_precedence)
* [Edit](https://en.cppreference.com/mwiki/index.php?title=cpp/language/operator_precedence&action=edit)
* [History](https://en.cppreference.com/mwiki/index.php?title=cpp/language/operator_precedence&action=history)

[](https://srv.carbonads.net/ads/click/x/GTND42JWCT7IP5QUF6BLYKQNCEYD6KQMFTBICZ3JCYBDTKJLCT7D62JKFTSD427UC67DT53LCW7IC237FTSI627KC6SDVK3ICEYDTK3EHJNCLSIZ?segment=placement:cppreferencecom;)[Limited time offer: Get 10 free Adobe Stock images.](https://srv.carbonads.net/ads/click/x/GTND42JWCT7IP5QUF6BLYKQNCEYD6KQMFTBICZ3JCYBDTKJLCT7D62JKFTSD427UC67DT53LCW7IC237FTSI627KC6SDVK3ICEYDTK3EHJNCLSIZ?segment=placement:cppreferencecom;)[**ADS VIA CARBON**](http://carbonads.net/?utm_source=cppreferencecom&utm_medium=ad_via_link&utm_campaign=in_unit&utm_term=carbon)

C++ Operator Precedence

[C++](https://en.cppreference.com/w/cpp)

[C++ language](https://en.cppreference.com/w/cpp/language)

[Expressions](https://en.cppreference.com/w/cpp/language/expressions)

The following table lists the precedence and associativity of C++ operators. Operators are listed top to bottom, in descending precedence.

|  |  |  |  |
| --- | --- | --- | --- |
| **Precedence** | **Operator** | **Description** | **Associativity** |
| **1** | :: | [Scope resolution](https://en.cppreference.com/w/cpp/language/identifiers#Qualified_identifiers) | Left-to-right |
| **2** | a++   a-- | Suffix/postfix [increment and decrement](https://en.cppreference.com/w/cpp/language/operator_incdec) |
| *type*()   *type*{} | [Functional cast](https://en.cppreference.com/w/cpp/language/explicit_cast) |
| a() | [Function call](https://en.cppreference.com/w/cpp/language/operator_other#Built-in_function_call_operator) |
| a[] | [Subscript](https://en.cppreference.com/w/cpp/language/operator_member_access#Built-in_subscript_operator) |
| .   -> | [Member access](https://en.cppreference.com/w/cpp/language/operator_member_access#Built-in_member_access_operators) |
| **3** | ++a   --a | Prefix [increment and decrement](https://en.cppreference.com/w/cpp/language/operator_incdec) | Right-to-left |
| +a   -a | Unary [plus and minus](https://en.cppreference.com/w/cpp/language/operator_arithmetic#Unary_arithmetic_operators) |
| !   ~ | [Logical NOT](https://en.cppreference.com/w/cpp/language/operator_logical) and [bitwise NOT](https://en.cppreference.com/w/cpp/language/operator_arithmetic#Bitwise_logic_operators) |
| (*type*) | [C-style cast](https://en.cppreference.com/w/cpp/language/explicit_cast) |
| \*a | [Indirection](https://en.cppreference.com/w/cpp/language/operator_member_access#Built-in_indirection_operator) (dereference) |
| &a | [Address-of](https://en.cppreference.com/w/cpp/language/operator_member_access#Built-in_address-of_operator) |
| sizeof | [Size-of](https://en.cppreference.com/w/cpp/language/sizeof)[[note 1]](https://en.cppreference.com/w/cpp/language/operator_precedence#cite_note-1) |
| co\_await | [await-expression](https://en.cppreference.com/w/cpp/language/coroutines) (C++20) |
| new   new[] | [Dynamic memory allocation](https://en.cppreference.com/w/cpp/language/new) |
| delete   delete[] | [Dynamic memory deallocation](https://en.cppreference.com/w/cpp/language/delete) |
| **4** | .\*   ->\* | [Pointer-to-member](https://en.cppreference.com/w/cpp/language/operator_member_access#Built-in_pointer-to-member_access_operators) | Left-to-right |
| **5** | a\*b   a/b   a%b | [Multiplication, division, and remainder](https://en.cppreference.com/w/cpp/language/operator_arithmetic#Multiplicative_operators) |
| **6** | a+b   a-b | [Addition and subtraction](https://en.cppreference.com/w/cpp/language/operator_arithmetic#Additive_operators) |
| **7** | <<   >> | Bitwise [left shift and right shift](https://en.cppreference.com/w/cpp/language/operator_arithmetic#Bitwise_shift_operators) |
| **8** | <=> | [Three-way comparison operator](https://en.cppreference.com/w/cpp/language/operator_comparison#Three-way_comparison) (since C++20) |
| **9** | <   <=   >   >= | For [relational operators](https://en.cppreference.com/w/cpp/language/operator_comparison) < and ≤ and > and ≥ respectively |
| **10** | ==   != | For [equality operators](https://en.cppreference.com/w/cpp/language/operator_comparison) = and ≠ respectively |
| **11** | & | [Bitwise AND](https://en.cppreference.com/w/cpp/language/operator_arithmetic#Bitwise_logic_operators) |
| **12** | ^ | [Bitwise XOR](https://en.cppreference.com/w/cpp/language/operator_arithmetic#Bitwise_logic_operators) (exclusive or) |
| **13** | | | [Bitwise OR](https://en.cppreference.com/w/cpp/language/operator_arithmetic#Bitwise_logic_operators) (inclusive or) |
| **14** | && | [Logical AND](https://en.cppreference.com/w/cpp/language/operator_logical) |
| **15** | || | [Logical OR](https://en.cppreference.com/w/cpp/language/operator_logical) |
| **16** | a?b:c | [Ternary conditional](https://en.cppreference.com/w/cpp/language/operator_other#Conditional_operator)[[note 2]](https://en.cppreference.com/w/cpp/language/operator_precedence#cite_note-2) | Right-to-left |
| throw | [throw operator](https://en.cppreference.com/w/cpp/language/throw) |
| co\_yield | [yield-expression](https://en.cppreference.com/w/cpp/language/coroutines) (C++20) |
| = | [Direct assignment](https://en.cppreference.com/w/cpp/language/operator_assignment#Builtin_direct_assignment) (provided by default for C++ classes) |
| +=   -= | [Compound assignment](https://en.cppreference.com/w/cpp/language/operator_assignment#Builtin_compound_assignment) by sum and difference |
| \*=   /=   %= | [Compound assignment](https://en.cppreference.com/w/cpp/language/operator_assignment#Builtin_compound_assignment) by product, quotient, and remainder |
| <<=   >>= | [Compound assignment](https://en.cppreference.com/w/cpp/language/operator_assignment#Builtin_compound_assignment) by bitwise left shift and right shift |
| &=   ^=   |= | [Compound assignment](https://en.cppreference.com/w/cpp/language/operator_assignment#Builtin_compound_assignment) by bitwise AND, XOR, and OR |
| **17** | , | [Comma](https://en.cppreference.com/w/cpp/language/operator_other#Built-in_comma_operator) | Left-to-right |

1. [↑](https://en.cppreference.com/w/cpp/language/operator_precedence#cite_ref-1) The operand of sizeof can't be a C-style type cast: the expression sizeof (int) \* p is unambiguously interpreted as (sizeof(int)) \* p, but not sizeof((int)\*p).
2. [↑](https://en.cppreference.com/w/cpp/language/operator_precedence#cite_ref-2) The expression in the middle of the conditional operator (between **?** and **:**) is parsed as if parenthesized: its precedence relative to **?:** is ignored.

When parsing an expression, an operator which is listed on some row of the table above with a precedence will be bound tighter (as if by parentheses) to its arguments than any operator that is listed on a row further below it with a lower precedence. For example, the expressions [std::cout](http://en.cppreference.com/w/cpp/io/cout) << a & b and \*p++ are parsed as ([std::cout](http://en.cppreference.com/w/cpp/io/cout) << a) & b and \*(p++), and not as [std::cout](http://en.cppreference.com/w/cpp/io/cout) << (a & b) or (\*p)++.

Operators that have the same precedence are bound to their arguments in the direction of their associativity. For example, the expression a = b = c is parsed as a = (b = c), and not as (a = b) = c because of right-to-left associativity of assignment, but a + b - c is parsed (a + b) - c and not a + (b - c) because of left-to-right associativity of addition and subtraction.

Associativity specification is redundant for unary operators and is only shown for completeness: unary prefix operators always associate right-to-left (delete ++\*p is delete(++(\*p))) and unary postfix operators always associate left-to-right (a[1][2]++ is ((a[1])[2])++). Note that the associativity is meaningful for member access operators, even though they are grouped with unary postfix operators: a.b++ is parsed (a.b)++ and not a.(b++).

Operator precedence is unaffected by [operator overloading](https://en.cppreference.com/w/cpp/language/operators). For example, [std::cout](http://en.cppreference.com/w/cpp/io/cout) << a ? b : c; parses as ([std::cout](http://en.cppreference.com/w/cpp/io/cout) << a) ? b : c; because the precedence of arithmetic left shift is higher than the conditional operator.

**Notes**

Precedence and associativity are compile-time concepts and are independent from [order of evaluation](https://en.cppreference.com/w/cpp/language/eval_order), which is a runtime concept.

The standard itself doesn't specify precedence levels. They are derived from the grammar.

[const\_cast](https://en.cppreference.com/w/cpp/language/const_cast), [static\_cast](https://en.cppreference.com/w/cpp/language/static_cast), [dynamic\_cast](https://en.cppreference.com/w/cpp/language/dynamic_cast), [reinterpret\_cast](https://en.cppreference.com/w/cpp/language/reinterpret_cast), [typeid](https://en.cppreference.com/w/cpp/language/typeid), [sizeof...](https://en.cppreference.com/w/cpp/language/sizeof...), [noexcept](https://en.cppreference.com/w/cpp/language/noexcept) and [alignof](https://en.cppreference.com/w/cpp/language/alignof) are not included since they are never ambiguous.

Some of the operators have [alternate spellings](https://en.cppreference.com/w/cpp/language/operator_alternative) (e.g., and for &&, or for ||, not for !, etc.).

In C, the ternary conditional operator has higher precedence than assignment operators. Therefore, the expression e = a < d ? a++ : a = d, which is parsed in C++ as e = ((a < d) ? (a++) : (a = d)), will fail to compile in C due to grammatical or semantic constraints in C. See the corresponding C page for details.

**See also**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Common operators** | | | | | | |
| [assignment](https://en.cppreference.com/w/cpp/language/operator_assignment) | [increment decrement](https://en.cppreference.com/w/cpp/language/operator_incdec) | [arithmetic](https://en.cppreference.com/w/cpp/language/operator_arithmetic) | [logical](https://en.cppreference.com/w/cpp/language/operator_logical) | [comparison](https://en.cppreference.com/w/cpp/language/operator_comparison) | [member access](https://en.cppreference.com/w/cpp/language/operator_member_access) | [other](https://en.cppreference.com/w/cpp/language/operator_other) |
| a = b a += b a -= b a \*= b a /= b a %= b a &= b a |= b a ^= b a <<= b a >>= b | ++a --a a++ a-- | +a -a a + b a - b a \* b a / b a % b ~a a & b a | b a ^ b a << b a >> b | !a a && b a || b | a == b a != b a < b a > b a <= b a >= b a <=> b | a[b] \*a &a a->b a.b a->\*b a.\*b | a(...) a, b ? : |
| **Special operators** | | | | | | |
| [static\_cast](https://en.cppreference.com/w/cpp/language/static_cast) converts one type to another related type [dynamic\_cast](https://en.cppreference.com/w/cpp/language/dynamic_cast) converts within inheritance hierarchies [const\_cast](https://en.cppreference.com/w/cpp/language/const_cast) adds or removes [cv](https://en.cppreference.com/w/cpp/language/cv) qualifiers [reinterpret\_cast](https://en.cppreference.com/w/cpp/language/reinterpret_cast) converts type to unrelated type [C-style cast](https://en.cppreference.com/w/cpp/language/explicit_cast) converts one type to another by a mix of static\_cast, const\_cast, and reinterpret\_cast [new](https://en.cppreference.com/w/cpp/language/new) creates objects with dynamic storage duration [delete](https://en.cppreference.com/w/cpp/language/delete) destructs objects previously created by the new expression and releases obtained memory area [sizeof](https://en.cppreference.com/w/cpp/language/sizeof) queries the size of a type [sizeof...](https://en.cppreference.com/w/cpp/language/sizeof...) queries the size of a [parameter pack](https://en.cppreference.com/w/cpp/language/parameter_pack) (since C++11) [typeid](https://en.cppreference.com/w/cpp/language/typeid) queries the type information of a type [noexcept](https://en.cppreference.com/w/cpp/language/noexcept) checks if an expression can throw an exception (since C++11) [alignof](https://en.cppreference.com/w/cpp/language/alignof) queries alignment requirements of a type (since C++11) | | | | | | |

|  |
| --- |
| [**C documentation**](https://en.cppreference.com/w/c/language/operator_precedence) for **C operator precedence** |

* [Support us](http://www.cppreference.com/support)
* [Recent changes](https://en.cppreference.com/w/Special:RecentChanges)
* [FAQ](https://en.cppreference.com/w/Cppreference:FAQ)
* [Offline version](https://en.cppreference.com/w/Cppreference:Archives)
* [What links here](https://en.cppreference.com/w/Special:WhatLinksHere/cpp/language/operator_precedence)
* [Related changes](https://en.cppreference.com/w/Special:RecentChangesLinked/cpp/language/operator_precedence)
* [Upload file](http://upload.cppreference.com/w/Special:Upload)
* [Special pages](https://en.cppreference.com/w/Special:SpecialPages)
* [Printable version](https://en.cppreference.com/mwiki/index.php?title=cpp/language/operator_precedence&printable=yes)
* [Permanent link](https://en.cppreference.com/mwiki/index.php?title=cpp/language/operator_precedence&oldid=131198)
* [Page information](https://en.cppreference.com/mwiki/index.php?title=cpp/language/operator_precedence&action=info)
* In other languages
* [العربية](http://ar.cppreference.com/w/cpp/language/operator_precedence)
* [Česky](http://cs.cppreference.com/w/cpp/language/operator_precedence)
* [Deutsch](http://de.cppreference.com/w/cpp/language/operator_precedence)
* [Español](http://es.cppreference.com/w/cpp/language/operator_precedence)
* [Français](http://fr.cppreference.com/w/cpp/language/operator_precedence)
* [Italiano](http://it.cppreference.com/w/cpp/language/operator_precedence)
* [日本語](http://ja.cppreference.com/w/cpp/language/operator_precedence)
* [한국어](http://ko.cppreference.com/w/cpp/language/operator_precedence)
* [Português](http://pt.cppreference.com/w/cpp/language/operator_precedence)
* [Русский](http://ru.cppreference.com/w/cpp/language/operator_precedence)
* [Türkçe](http://tr.cppreference.com/w/cpp/language/operator_precedence)
* [中文](http://zh.cppreference.com/w/cpp/language/operator_precedence)
* This page was last modified on 6 July 2021, at 17:55.
* [Privacy policy](https://en.cppreference.com/w/Cppreference:Privacy_policy)

* [About cppreference.com](https://en.cppreference.com/w/Cppreference:About)

* [Disclaimers](https://en.cppreference.com/w/Cppreference:General_disclaimer)
* [Powered by MediaWiki](https://www.mediawiki.org/) [Powered by GeSHi](http://qbnz.com/highlighter/) [Hosted by Tiger Technologies](http://www.tigertech.net/referral/cppreference.com)