|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Operator name** | | **Syntax** | **Can overload in C++** | **Included in**[**C**](https://en.wikipedia.org/wiki/C_(programming_language)) | **C++ Prototype examples** | |
| **As member of K** | **Outside class definitions** |
| [Basic assignment](https://en.wikipedia.org/wiki/Assignment_operator_in_C%2B%2B) | | a **=** b | Yes | Yes | R& K::**operator** =(S b); | N/A |
| [Addition](https://en.wikipedia.org/wiki/Addition) | | a **+** b | Yes | Yes | R K::**operator** +(S b); | R **operator** +(K a, S b); |
| [Subtraction](https://en.wikipedia.org/wiki/Subtraction) | | a **-** b | Yes | Yes | R K::**operator** -(S b); | R **operator** -(K a, S b); |
| [Unary](https://en.wikipedia.org/wiki/Unary_operation) plus ([integer promotion](https://en.wikipedia.org/wiki/Type_conversion#Type_promotion)) | | **+**a | Yes | Yes | R K::**operator** +(); | R **operator** +(K a); |
| Unary minus ([additive inverse](https://en.wikipedia.org/wiki/Additive_inverse)) | | **-**a | Yes | Yes | R K::**operator** -(); | R **operator** -(K a); |
| [Multiplication](https://en.wikipedia.org/wiki/Multiplication) | | a **\*** b | Yes | Yes | R K::**operator** \*(S b); | R **operator** \*(K a, S b); |
| [Division](https://en.wikipedia.org/wiki/Division_(mathematics)) | | a **/** b | Yes | Yes | R K::**operator** /(S b); | R **operator** /(K a, S b); |
| [Modulo](https://en.wikipedia.org/wiki/Modulo_operation) (integer remainder)[[a]](https://en.wikipedia.org/wiki/Operators_in_C_and_C%2B%2B#cite_note-modulo-1) | | a **%** b | Yes | Yes | R K::**operator** %(S b); | R **operator** %(K a, S b); |
| [Increment](https://en.wikipedia.org/wiki/Increment_and_decrement_operators) | Prefix | **++**a | Yes | Yes | R& K::**operator** ++(); | R& **operator** ++(K& a); |
| Postfix | a**++** | Yes | Yes | R K::**operator** ++(int); | R **operator** ++(K& a, int); |
| Note: [C++](https://en.wikipedia.org/wiki/C%2B%2B) uses the unnamed dummy-parameter int to differentiate between prefix and postfix increment operators. | |
| [Decrement](https://en.wikipedia.org/wiki/Increment_and_decrement_operators) | Prefix | **--**a | Yes | Yes | R& K::**operator** --(); | R& **operator** --(K& a); |
| Postfix | a**--** | Yes | Yes | R K::**operator** --(int); | R **operator** --(K& a, int); |
| Note: [C++](https://en.wikipedia.org/wiki/C%2B%2B) uses the unnamed dummy-parameter int to differentiate between prefix and postfix decrement operators. | |

**Comparison operators/relational operators**[[edit](https://en.wikipedia.org/w/index.php?title=Operators_in_C_and_C%2B%2B&action=edit&section=3)]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Operator name** | **Syntax** | **Can overload in C++** | **Included in**[**C**](https://en.wikipedia.org/wiki/C_(programming_language)) | **Prototype examples** | |
| **As member of K** | **Outside class definitions** |
| Equal to | a **==** b | Yes | Yes | bool K::**operator** ==(S **const**&b); | bool **operator** ==(K **const**& a, S**const**& b); |
| Not equal to | a **!=** b a **not\_eq** b[[b]](https://en.wikipedia.org/wiki/Operators_in_C_and_C%2B%2B#cite_note-synonyms-2) | Yes | Yes | bool K::**operator** !=(S **const**&b); bool K::**operator** !=(S**const**& b) **const**; | bool **operator** !=(K **const**& a, S**const**& b); |
| Greater than | a **>** b | Yes | Yes | bool K::**operator** >(S **const**& b)**const**; | bool **operator** >(K **const**& a, S**const**& b); |
| Less than | a **<** b | Yes | Yes | bool K::**operator** <(S **const**&b)**const**; | bool **operator** <(K **const**& a, S**const**& b); |
| Greater than or equal to | a **>=** b | Yes | Yes | bool K::**operator** >=(S **const**& b)**const**; | bool **operator** >=(K **const**& a, S**const**& b); |
| Less than or equal to | a **<=** b | Yes | Yes | bool K::**operator** <=(S **const**&b); | bool **operator** <=(K **const**& a, S**const**& b); |

**Logical operators**[[edit](https://en.wikipedia.org/w/index.php?title=Operators_in_C_and_C%2B%2B&action=edit&section=4)]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Operator name** | **Syntax** | **Can overload in C++** | **Included in**[**C**](https://en.wikipedia.org/wiki/C_(programming_language)) | **Prototype examples** | |
| **As member of K** | **Outside class definitions** |
| [Logical negation (NOT)](https://en.wikipedia.org/wiki/Negation) | **!**a **not** a[[b]](https://en.wikipedia.org/wiki/Operators_in_C_and_C%2B%2B#cite_note-synonyms-2) | Yes | Yes | bool K::**operator** !(); | bool **operator** !(K a); |
| [Logical AND](https://en.wikipedia.org/wiki/Logical_conjunction) | a **&&** b a **and** b[[b]](https://en.wikipedia.org/wiki/Operators_in_C_and_C%2B%2B#cite_note-synonyms-2) | Yes | Yes | bool K::**operator** &&(S b); | bool **operator** &&(K a, S b); |
| [Logical OR](https://en.wikipedia.org/wiki/Logical_disjunction) | a **||** b a **or** b[[b]](https://en.wikipedia.org/wiki/Operators_in_C_and_C%2B%2B#cite_note-synonyms-2) | Yes | Yes | bool K::**operator** ||(S b); | bool **operator** ||(K a, S b); |

**Bitwise operators**[[edit](https://en.wikipedia.org/w/index.php?title=Operators_in_C_and_C%2B%2B&action=edit&section=5)]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Operator name** | **Syntax** | **Can overload in C++** | **Included in**[**C**](https://en.wikipedia.org/wiki/C_(programming_language)) | **Prototype examples** | |
| **As member of K** | **Outside class definitions** |
| [Bitwise NOT](https://en.wikipedia.org/wiki/Bitwise_operation#NOT) | **~**a **compl** a[[b]](https://en.wikipedia.org/wiki/Operators_in_C_and_C%2B%2B#cite_note-synonyms-2) | Yes | Yes | R K::**operator** ~(); | R **operator** ~(K a); |
| [Bitwise AND](https://en.wikipedia.org/wiki/Bitwise_operation#AND) | a **&** b a **bitand** b[[b]](https://en.wikipedia.org/wiki/Operators_in_C_and_C%2B%2B#cite_note-synonyms-2) | Yes | Yes | R K::**operator** &(S b); | R **operator** &(K a, S b); |
| [Bitwise OR](https://en.wikipedia.org/wiki/Bitwise_operation#OR) | a **|** b a **bitor** b[[b]](https://en.wikipedia.org/wiki/Operators_in_C_and_C%2B%2B#cite_note-synonyms-2) | Yes | Yes | R K::**operator** |(S b); | R **operator** |(K a, S b); |
| [Bitwise XOR](https://en.wikipedia.org/wiki/Bitwise_operation#XOR) | a **^** b a **xor** b[[b]](https://en.wikipedia.org/wiki/Operators_in_C_and_C%2B%2B#cite_note-synonyms-2) | Yes | Yes | R K::**operator** ^(S b); | R **operator** ^(K a, S b); |
| [Bitwise left shift](https://en.wikipedia.org/wiki/Bitwise_shift)[[c]](https://en.wikipedia.org/wiki/Operators_in_C_and_C%2B%2B#cite_note-bitshift-3) | a **<<** b | Yes | Yes | R K::**operator** <<(S b); | R **operator** <<(K a, S b); |
| [Bitwise right shift](https://en.wikipedia.org/wiki/Bitwise_shift)[[c]](https://en.wikipedia.org/wiki/Operators_in_C_and_C%2B%2B#cite_note-bitshift-3)[[d]](https://en.wikipedia.org/wiki/Operators_in_C_and_C%2B%2B#cite_note-rightbitshift-5) | a **>>** b | Yes | Yes | R K::**operator** >>(S b); | R **operator** >>(K a, S b); |

**Compound assignment operators**[[edit](https://en.wikipedia.org/w/index.php?title=Operators_in_C_and_C%2B%2B&action=edit&section=6)]

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Operator name** | **Syntax** | **Meaning** | **Can overload in C++** | **Included in**[**C**](https://en.wikipedia.org/wiki/C_(programming_language)) | **Prototype examples** | |
| **As member of K** | **Outside class definitions** |
| Addition assignment | a **+=** b | a **=** a **+** b | Yes | Yes | R& K::**operator** +=(S b); | R& **operator** +=(K& a, S b); |
| Subtraction assignment | a **-=** b | a **=** a **-** b | Yes | Yes | R& K::**operator** -=(S b); | R& **operator** -=(K& a, S b); |
| Multiplication assignment | a **\*=** b | a **=** a **\*** b | Yes | Yes | R& K::**operator** \*=(S b); | R& **operator** \*=(K& a, S b); |
| Division assignment | a **/=** b | a **=** a **/** b | Yes | Yes | R& K::**operator** /=(S b); | R& **operator** /=(K& a, S b); |
| Modulo assignment | a **%=** b | a **=** a **%** b | Yes | Yes | R& K::**operator** %=(S b); | R& **operator** %=(K& a, S b); |
| Bitwise AND assignment | a **&=** b a **and\_eq** b[[b]](https://en.wikipedia.org/wiki/Operators_in_C_and_C%2B%2B#cite_note-synonyms-2) | a **=** a **&** b | Yes | Yes | R& K::**operator** &=(S b); | R& **operator** &=(K& a, S b); |
| Bitwise OR assignment | a **|=** b a **or\_eq** b[[b]](https://en.wikipedia.org/wiki/Operators_in_C_and_C%2B%2B#cite_note-synonyms-2) | a **=** a **|** b | Yes | Yes | R& K::**operator** |=(S b); | R& **operator** |=(K& a, S b); |
| Bitwise XOR assignment | a **^=** b a **xor\_eq** b[[b]](https://en.wikipedia.org/wiki/Operators_in_C_and_C%2B%2B#cite_note-synonyms-2) | a **=** a **^** b | Yes | Yes | R& K::**operator** ^=(S b); | R& **operator** ^=(K& a, S b); |
| Bitwise left shift assignment | a **<<=** b | a **=** a **<<** b | Yes | Yes | R& K::**operator** <<=(S b); | R& **operator** <<=(K& a, S b); |
| Bitwise right shift assignment[[d]](https://en.wikipedia.org/wiki/Operators_in_C_and_C%2B%2B#cite_note-rightbitshift-5) | a **>>=** b | a **=** a **>>** b | Yes | Yes | R& K::**operator** >>=(S b); | R& **operator** >>=(K& a, S b); |

**Member and pointer operators**[[edit](https://en.wikipedia.org/w/index.php?title=Operators_in_C_and_C%2B%2B&action=edit&section=7)]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Operator name** | **Syntax** | **Can overload in C++** | **Included in**[**C**](https://en.wikipedia.org/wiki/C_(programming_language)) | **Prototype examples** | |
| **As member of K** | **Outside class definitions** |
| [Subscript](https://en.wikipedia.org/wiki/Indexer_(programming)) | a**[**b**]** | Yes | Yes | R& K::**operator** [](S b); | N/A |
| Indirection ("object pointed to by *a*") | **\***a | Yes | Yes | R& K::**operator** \*(); | R& **operator** \*(K a); |
| Address-of ("address of *a*") | **&**a | Yes | Yes | R\* K::**operator** &(); | R\* **operator** &(K a); |
| Structure dereference ("member *b*of object pointed to by *a*") | a**->**b | Yes | Yes | R\* K::**operator** ->();[[e]](https://en.wikipedia.org/wiki/Operators_in_C_and_C%2B%2B#cite_note-arrowptr-6) | N/A |
| Structure reference ("member *b* of object *a*") | a**.**b | No | Yes | N/A | |
| Member selected by [pointer-to-member](https://en.wikipedia.org/w/index.php?title=Pointer-to-member&action=edit&redlink=1) *b* of object pointed to by *a*[[f]](https://en.wikipedia.org/wiki/Operators_in_C_and_C%2B%2B#cite_note-arrowstar-7) | a**->\***b | Yes | No | R& K::**operator** ->\*(S b); | R& **operator** ->\*(K a, S b); |
| Member of object *a* selected by [pointer-to-member](https://en.wikipedia.org/w/index.php?title=Pointer-to-member&action=edit&redlink=1) *b* | a**.\***b | No | No | N/A | |

**Other operators**[[edit](https://en.wikipedia.org/w/index.php?title=Operators_in_C_and_C%2B%2B&action=edit&section=8)]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Operator name** | **Syntax** | **Can overload in C++** | **Included in**[**C**](https://en.wikipedia.org/wiki/C_(programming_language)) | **Prototype examples** | |
| **As member of K** | **Outside class definitions** |
| [Function](https://en.wikipedia.org/wiki/Function_(programming)) call *See*[*Function object*](https://en.wikipedia.org/wiki/Function_object). | a**(**a1, a2**)** | Yes | Yes | R K::**operator** ()(S a, T b,...); | N/A |
| [Comma](https://en.wikipedia.org/wiki/Comma_operator) | a**,** b | Yes | Yes | R K::**operator** ,(S b); | R **operator** ,(K a, S b); |
| [Ternary conditional](https://en.wikipedia.org/wiki/%3F:) | a **?** b **:** c | No | Yes | N/A | |
| [Scope resolution](https://en.wikipedia.org/wiki/Scope_resolution_operator#C++) | a**::**b | No | No | N/A | |
| User-defined literals[[g]](https://en.wikipedia.org/wiki/Operators_in_C_and_C%2B%2B#cite_note-ud-literal-8) *since C++11* | "a"\_b | Yes | No | N/A | R **operator** "" \_b(T a) |
| [Size-of](https://en.wikipedia.org/wiki/Sizeof) | **sizeof** (a)[[h]](https://en.wikipedia.org/wiki/Operators_in_C_and_C%2B%2B#cite_note-sizeof-9) **sizeof** (*type*) | No | Yes | N/A | |
| Size of [parameter pack](https://en.wikipedia.org/wiki/Variadic_template) *since C++11* | **sizeof...**(Args) | No | No | N/A | |
| [Align-of](https://en.wikipedia.org/w/index.php?title=Alignof&action=edit&redlink=1) *since C++11* | **alignof** (*type*) or **\_Alignof** (*type*)[[i]](https://en.wikipedia.org/wiki/Operators_in_C_and_C%2B%2B#cite_note-alignof-10) | No | Yes | N/A | |
| Type identification | **typeid** (a) **typeid** (*type*) | No | No | N/A | |
| [Conversion](https://en.wikipedia.org/wiki/Type_conversion) (C-style cast) | (*type*) a | No | Yes | N/A | |
| [Conversion](https://en.wikipedia.org/wiki/Type_conversion) | *type*(a) | No | No | Note: behaves like const\_cast/static\_cast/reinterpret\_cast[[2]](https://en.wikipedia.org/wiki/Operators_in_C_and_C%2B%2B#cite_note-11) | |
| [static\_cast](https://en.wikipedia.org/wiki/Static_cast)conversion | **static\_cast**<*type*>(a) | Yes | No | K::**operator** R(); **explicit** K::**operator** R(); *since C++11* | N/A |
| Note: for user-defined conversions, the return type implicitly and necessarily matches the operator name. | |
| [dynamic cast](https://en.wikipedia.org/wiki/Dynamic_cast)conversion | **dynamic\_cast**<*type*>(a) | No | No | N/A | |
| [const\_cast](https://en.wikipedia.org/wiki/Const_cast)conversion | **const\_cast**<*type*>(a) | No | No | N/A | |
| [reinterpret\_cast](https://en.wikipedia.org/wiki/Reinterpret_cast)conversion | **reinterpret\_cast**<*type*>(a) | No | No | N/A | |
| [Allocate storage](https://en.wikipedia.org/wiki/New_(c%2B%2B)) | **new** *type* | Yes | No | void\* K::**operator** **new**(size\_tx); | void\* **operator** new(size\_t x); |
| Allocate storage (array) | **new** *type***[**n**]** | Yes | No | void\* K::**operator** **new**[](size\_ta); | void\* **operator** **new**[](size\_t a); |
| [Deallocate storage](https://en.wikipedia.org/wiki/Operator_delete) | **delete** a | Yes | No | void K::**operator** **delete**(void\*a); | void **operator** delete(void \*a); |
| Deallocate storage (array) | **delete[]** a | Yes | No | void K::**operator** **delete**[](void\*a); | void **operator** **delete**[](void\*a); |
| Exception check *since C++11* | **noexcept**(a) | No | No | N/A | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Precedence** | **Operator** | **Description** | **Associativity** |
| **1**  **highest** | :: | [Scope resolution](https://en.wikipedia.org/wiki/Scope_resolution_operator#C.2B.2B) (C++ only) | None |
| **2** | ++ | Postfix increment | Left-to-right |
| -- | Postfix decrement |
| () | Function call |
| [] | Array subscripting |
| . | Element selection by reference |
| -> | Element selection through pointer |
| typeid() | [Run-time type information](https://en.wikipedia.org/wiki/Run-time_type_information) (C++ only) (see [typeid](https://en.wikipedia.org/wiki/Typeid)) |
| const\_cast | Type cast (C++ only) (see [const\_cast](https://en.wikipedia.org/wiki/Const_cast)) |
| dynamic\_cast | Type cast (C++ only) (see [dynamic cast](https://en.wikipedia.org/wiki/Dynamic_cast)) |
| reinterpret\_cast | Type cast (C++ only) (see [reinterpret\_cast](https://en.wikipedia.org/wiki/Reinterpret_cast)) |
| static\_cast | Type cast (C++ only) (see [static\_cast](https://en.wikipedia.org/wiki/Static_cast)) |
| **3** | ++ | Prefix increment | Right-to-left |
| -- | Prefix decrement |
| + | Unary plus |
| - | Unary minus |
| ! | Logical NOT |
| ~ | Bitwise NOT (One's Complement) |
| (*type*) | Type cast |
| \* | Indirection (dereference) |
| & | Address-of |
| sizeof | [Size-of](https://en.wikipedia.org/wiki/Sizeof) |
| new, new[] | Dynamic memory allocation (C++ only) |
| delete, delete[] | Dynamic memory deallocation (C++ only) |
| **4** | .\* | Pointer to member (C++ only) | Left-to-right |
| ->\* | Pointer to member (C++ only) |
| **5** | \* | Multiplication | Left-to-right |
| / | Division |
| % | [Modulo](https://en.wikipedia.org/wiki/Modulo_operation) (remainder) |
| **6** | + | Addition | Left-to-right |
| - | Subtraction |
| **7** | << | [Bitwise](https://en.wikipedia.org/wiki/Bitwise_operation) left shift | Left-to-right |
| >> | [Bitwise](https://en.wikipedia.org/wiki/Bitwise_operation) right shift |
| **8** | < | Less than | Left-to-right |
| <= | Less than or equal to |
| > | Greater than |
| >= | Greater than or equal to |
| **9** | == | Equal to | Left-to-right |
| != | Not equal to |
| **10** | & | Bitwise AND | Left-to-right |
| **11** | ^ | Bitwise XOR (exclusive or) | Left-to-right |
| **12** | | | Bitwise OR (inclusive or) | Left-to-right |
| **13** | && | Logical AND | Left-to-right |
| **14** | || | Logical OR | Left-to-right |
| **15** | ?: | [Ternary](https://en.wikipedia.org/wiki/Ternary_operator) conditional (see [?:](https://en.wikipedia.org/wiki/%3F:)) | Right-to-left |
| **16** | = | Direct assignment | Right-to-left |
| += | Assignment by sum |
| -= | Assignment by difference |
| \*= | Assignment by product |
| /= | Assignment by quotient |
| %= | Assignment by remainder |
| <<= | Assignment by bitwise left shift |
| >>= | Assignment by bitwise right shift |
| &= | Assignment by bitwise AND |
| ^= | Assignment by bitwise XOR |
| |= | Assignment by bitwise OR |
| **17** | throw | Throw operator (exceptions throwing, C++ only) | Right-to-left |
| **18**  **lowest** | , | [Comma](https://en.wikipedia.org/wiki/Comma_operator) | Left-to-right |