

Anexo:Operadores de C y C++

Esta es una lista de los **operadores de los lenguajes de programación C y C++**. Todos los operadores listados existen en C++. La tercera columna indica si también están presentes en C. También hay que tener en cuenta que C no permite la sobrecarga de operadores.

Los siguientes operadores son puntos de secuencia en ambos lenguajes (cuando no están sobrecargados): `&&`, `||`, `?:`, y `,` (el operador coma).

C++ también incluye los operadores de conversión de tipos `const_cast`, `static_cast`, `dynamic_cast` y `reinterpret_cast`, que no están listados en la tabla por brevedad. El formato de estos operadores significa que su nivel de precedencia no es importante.

La mayoría de los operadores presentes en C y C++ (con la excepción del operador coma y el operador flecha) también se encuentran disponibles en los lenguajes de programación Java, Perl, C# y PHP con la misma precedencia, asociatividad y semántica. Con una única excepción: la asociatividad del operador ternario en PHP es de izquierda a derecha.

Tabla

Para los objetivos de esta tabla `a`, `b` y `c` representan valores válidos (literales, valores de variables o valores de retorno), nombres de objetos o lvalores según el caso.





Operadores aritméticos			
Nombre del operador	Sintaxis	Sobrecargable	Incluido en C
Más unitario	<code>+a</code>	✓ Sí	✓ Sí
Suma	<code>a + b</code>	✓ Sí	✓ Sí
Preincremento	<code>++a</code>	✓ Sí	✓ Sí
Postincremento	<code>a++</code>	✓ Sí	✓ Sí
Asignación con suma	<code>a += b</code>	✓ Sí	✓ Sí
Menos unitario (negación)	<code>-a</code>	✓ Sí	✓ Sí
Resta	<code>a - b</code>	✓ Sí	✓ Sí
Predecremento	<code>--a</code>	✓ Sí	✓ Sí
Postdecremento	<code>a--</code>	✓ Sí	✓ Sí
Asignación con resta	<code>a -= b</code>	✓ Sí	✓ Sí
Multiplicación	<code>a * b</code>	✓ Sí	✓ Sí
Asignación con multiplicación	<code>a *= b</code>	✓ Sí	✓ Sí
División	<code>a / b</code>	✓ Sí	✓ Sí
Asignación con división	<code>a /= b</code>	✓ Sí	✓ Sí
Módulo (Resto)	<code>a % b</code>	✓ Sí	✓ Sí
Asignación con módulo	<code>a %= b</code>	✓ Sí	✓ Sí
Operadores de comparación			
Nombre del operador	Sintaxis	Sobrecargable	Incluido en C
Menor que	<code>a < b</code>	✓ Sí	✓ Sí
Menor o igual que	<code>a <= b</code>	✓ Sí	✓ Sí













Mayor que	<code>a > b</code>	✓ Sí	✓ Sí
Mayor o igual que	<code>a >= b</code>	✓ Sí	✓ Sí
No igual que	<code>a != b</code>	✓ Sí	✓ Sí
Igual que	<code>a == b</code>	✓ Sí	✓ Sí
Negación lógica	<code>!a</code>	✓ Sí	✓ Sí
AND lógico	<code>a && b</code>	✓ Sí	✓ Sí
OR lógico	<code>a b</code>	✓ Sí	✓ Sí

Operadores a nivel de bit





Nombre del operador	Sintaxis	Sobrecargable	Incluido en C
Desplazamiento a la izquierda	<code>a << b</code>	✓ Sí	✓ Sí
Asignación con desplazamiento a la izquierda	<code>a <<= b</code>	✓ Sí	✓ Sí
Desplazamiento a la derecha	<code>a >> b</code>	✓ Sí	✓ Sí
Asignación con desplazamiento a la derecha	<code>a >>= b</code>	✓ Sí	✓ Sí
Complemento a uno	<code>~a</code>	✓ Sí	✓ Sí
AND binario	<code>a & b</code>	✓ Sí	✓ Sí
Asignación con AND binario	<code>a &= b</code>	✓ Sí	✓ Sí
OR binario	<code>a b</code>	✓ Sí	✓ Sí
Asignación con OR binario	<code>a = b</code>	✓ Sí	✓ Sí
XOR binario	<code>a ^ b</code>	✓ Sí	✓ Sí
Asignación con XOR binario	<code>a ^= b</code>	✓ Sí	✓ Sí

Otros operadores

Nombre del operador	Sintaxis	Sobrecargable	Incluido en C
Asignación básica	<code>a = b</code>	✓ Sí	✓ Sí
Llamada a función	<code>a ()</code>	✓ Sí	✓ Sí
Índice de Array	<code>a [b]</code>	✓ Sí	✓ Sí
Indirección (Desreferencia)	<code>*a</code>	✓ Sí	✓ Sí
Dirección de (Referencia)	<code>&a</code>	✓ Sí	✓ Sí
Miembro de puntero	<code>a->b</code>	✓ Sí	✓ Sí
Miembro	<code>a.b</code>	 No	✓ Sí
Desreferencia a miembro por puntero	<code>a->*b</code>	✓ Sí	 No
Desreferencia a miembro por objeto	<code>a.*b</code>	 No	 No
Conversión de tipo	<code>(tipo) a</code>	✓ Sí	✓ Sí

Coma	<code>a , b</code>	✓ Sí	✓ Sí
Condicional ternario	<code>a ? b : c</code>	 No	✓ Sí
Resolución de ámbito	<code>a::b</code>	 No	 No
Puntero a función miembro	<code>a::*b</code>	 No	 No
Tamaño de	<code>sizeof a</code> <code>sizeof(tipo)</code>	 No	✓ Sí
Identificación de tipo	<code>typeid(a)</code> <code>typeid(tipo)</code>	 No	 No
Asignar almacenamiento	<code>new tipo</code>	✓ Sí	 No
Asignar almacenamiento (Vector)	<code>new tipo[n]</code>	✓ Sí	 No
Desasignar almacenamiento	<code>delete a</code>	✓ Sí	 No
Desasignar almacenamiento (Vector)	<code>delete[] a</code>	✓ Sí	 No

Extensiones del lenguaje

Nombre del operador	Sintaxis	Sobrecargable	Incluido en C	Compilador
Dirección de la etiqueta	<code>&& etiqueta</code>	 No	✓ Sí	GCC / G++
Obtener tipo	<code>typeof a</code> <code>typeof (expr)</code>	 No	✓ Sí	GCC / G++
min y max	<code>a <? b</code> <code>a >? b</code>	 No	 No	G++

Precedencia de operadores

La tabla siguiente es una lista que muestra el orden de precedencia y la asociatividad de todos los operadores del lenguaje de programación C++. Están listados de arriba a abajo por orden de precedencia descendente y con la misma descendencia en la misma celda (puede haber varias filas de operadores en la misma celda). La precedencia de los operadores no cambia por la sobrecarga.

Una tabla de precedencias, aunque adecuada, no puede resolver todos los detalles. Por ejemplo, el operador ternario permite expresiones arbitrarias como operador central independientemente de la precedencia del resto de operadores. Así `a ? b , c : d` es interpretado como `a ? (b, c) : d` en vez de `(a ? b) , (c : d)`. También hay que tener en cuenta que el resultado sin paréntesis de una expresión de conversión en C no puede ser el operando de `sizeof`. Por eso `sizeof (int) * x` es interpretado como `(sizeof(int)) * x` y no como `sizeof ((int) *x)`.

Operador	Descripción	Asociatividad
<code>::</code>	Resolución de ámbito (solo C++)	Izquierda a derecha
<code>++ --</code> <code>()</code> <code>[]</code> <code>.</code> <code>-></code> <code>typeid()</code> <code>const_cast</code> <code>dynamic_cast</code> <code>reinterpret_cast</code> <code>static_cast</code>	Post- incremento y decremento Llamada a función Elemento de vector Selección de elemento por referencia Selección de elemento con puntero Información de tipo en tiempo de ejecución (solo C++) Conversión de tipo (solo C++) Conversión de tipo (solo C++) Conversión de tipo (solo C++) Conversión de tipo (solo C++)	
<code>++ --</code> <code>+ -</code> <code>! ~</code> <code>(type)</code> <code>*</code> <code>&</code> <code>sizeof</code> <code>new new[]</code> <code>delete delete[]</code>	Pre- incremento y decremento Suma y resta unitaria NOT lógico y NOT binario Conversión de tipo Indirección Dirección de Tamaño de Asignación dinámica de memoria (solo C++) Desasignación dinámica de memoria (solo C++)	Derecha a izquierda

<code>. * -> *</code>	Puntero a miembro (solo C++)	Izquierda a derecha
<code>* / %</code>	Multiplicación, división y módulo	
<code>+ -</code>	Suma y resta	
<code><< >></code>	Operaciones binarias de desplazamiento	
<code>< <=</code> <code>> >=</code>	Operadores relaciones "menor que", "menor o igual que", "mayor que" y "mayor o igual que"	
<code>== !=</code>	Operadores relaciones "igual a" y "distinto de"	
<code>&</code>	AND binario	
<code>^</code>	XOR binario	
<code> </code>	OR binario	
<code>& &</code>	AND lógico	
<code> </code>	OR lógico	
<code>c ? t : f</code>	Operador ternario	Derecha a izquierda
<code>=</code> <code>+= -=</code> <code>*= /= %=</code> <code><<= >>=</code> <code>&= ^= =</code>	Asignaciones	
<code>throw</code>	Operador Throw (lanzamiento de excepciones, solo C++)	
<code>,</code>	Coma	Izquierda a derecha

Fuentes y contribuyentes del artículo

Anexo:Operadores de C y C++ *Fuente:* <http://es.wikipedia.org/w/index.php?oldid=66259236> *Contribuyentes:* BetoCG, Biasoli, Diegusjaimes, Gelo71, GermanX, Halfdrag, Jkbw, Jrobertiko, Juan Antonio Cordero, Karras, Kroji, Lucas.lucas.lucas24, Matdrones, Richy, WikiCholi, 59 ediciones anónimas

Fuentes de imagen, Licencias y contribuyentes

Archivo:Yes_check.svg *Fuente:* http://es.wikipedia.org/w/index.php?title=Archivo:Yes_check.svg *Licencia:* Public Domain *Contribuyentes:* SVG by Gregory Maxwell (modified by WarX)

Archivo:X mark.svg *Fuente:* http://es.wikipedia.org/w/index.php?title=Archivo:X_mark.svg *Licencia:* Public Domain *Contribuyentes:* User:Gmaxwell

Licencia

Creative Commons Attribution-Share Alike 3.0 Unported
[//creativecommons.org/licenses/by-sa/3.0/](http://creativecommons.org/licenses/by-sa/3.0/)
