Operadores en Lenguaje C++

Tipos de operadores

- Aritméticos
- Relacionales
- De asignación
- De incremento y decremento
- Lógicos

Operadores aritméticos y nivel de precedencia

Operador(es)	Operación(es)	Orden de evaluación (precedencia)
`( )`	Paréntesis	Se evalúa primero. Si los paréntesis son *anidados*, como en la expresión `(a * ( b + c / d + e ) )`, la expresión en el par *más interno* se evalúa primero. [*Precaución:* si tiene una expresión como `(a + b) * (c - d)` en donde dos conjuntos de paréntesis no están anidados, pero aparecen "en el mismo nivel", en C++ estándar *no* no especifica el orden en el que se evaluarán estas subexpresiones entre paréntesis].

`*`
`/`
`%`
	Multiplicación
	División
	Módulo
		Se evalúan en segundo lugar. Si hay varios operadores de este tipo, se evalúan de izquierda a derecha.

`+` `-`	Suma Resta	Se evalúan al último. Si hay varios operadores de este tipo, se evalúan de izquierda a derecha.

Fuente: Como programar en C++, Deitel & Deitel, 9 Edi

Ejemplos de Operadores aritméticos

Ejemplo 1:

10 % 2 + 3 * 2 / 2 + 10 * 2 - 3

0 + 3 * 2 / 2 + 10 * 2 - 3

0 + 6 / 2 + 10 * 2 - 3

0 + 3 + 10 * 2 - 3

0 + 3 + 20 – 3

3 + 20 - 3

23 - 3

20

Ejemplo 2:

10 % (2 + 3) * 2 / 2 + 10 * (2 - 3)

10 % 5 * 2 / 2 + 10 * -1

0 * 2 / 2 + 10 * -1

0 / 2 + 10 * -1

0 + 10 * -1

0 + -10

-10

Ejemplos de Operadores aritméticos

Ejemplo 3:

5 + pow(2, 3) * 3 + sqrt(2) - 50 * 2

5 + 8 * 3 + sqrt(2) - 50 * 2

5 + 8 * 3 + 1.4142 - 50 * 2

5 + 24 + 1.4142 - 50 * 2

5 + 24 + 1.4142 - 100

29 + 1.4142 - 100

30.4142 - 100

-69.5858

Aunque **pow** y **sqrt** no son operadores aritméticos, tienen mayor precedencia al ser funciones matemáticas

Operadores relacionales

Son operadores binarios, es decir actúan sobre dos operandos, uno a la izquierda y otro a la derecha.
Además son operadores lógicos, ya que retornan un valor lógico: ***true*** o ***false***.

Operador algebraico de igualdad o relacional	Operador de igualdad o relacional de C++	Ejemplo de condición en C++	Significado de la condición en C++
Operadores relacionales			
>	>	x > y	x es mayor que y
<	<	x < y	x es menor que y
≥	>=	x >= y	x es mayor o igual que y
≤	<=	x <= y	x es menor o igual que y
Operadores de igualdad			
=	==	x == y	x es igual a y
≠	!=	x != y	x no es igual a y

Fuente: Como programar en C++, Deitel & Deitel, 9 Edi

Ejemplos de Operadores relacionales

Ejemplo:

5 > 10	***false***
5 < 10	***true***
5 == 5	***true***
5 != 5	***false***
2 >= 2	***true***
2 <= 2	***true***
2 > 2	***false***

Operadores de asignación

C++ cuenta con varios operadores de asignación para abreviar las expresiones de asignación.

Operador de asignación	Expresión de ejemplo	Explicación	Asigna
Suponer que: `int c = 3, d = 5, e = 4, f = 6, g = 12;`			
`+=`	`c += 7`	`c = c + 7`	`10 a c`
`-=`	`d -= 4`	`d = d - 4`	`1 a d`
`*=`	`e *= 5`	`e = e * 5`	`20 a e`
`/=`	`f /= 3`	`f = f / 3`	`2 a f`
`%=`	`g %= 9`	`g = g % 9`	`3 a g`

Fuente: Como programar en C++, Deitel & Deitel, 9 Edi

Operadores de incremento y decremento

Además de los operadores de asignación aritméticos, C++ proporciona dos operadores unarios para sumar 1, o restar 1, al valor de una variable numérica. Estos son el operador de incremento unario ++, y el operador de decremento unario --

Operador	Llamado	Expresión de ejemplo	Explicación
++	preincremento	++a	Incrementar a en 1, después utilizar el nuevo valor de a en la expresión en que esta variable reside.
++	postincremento	a++	Usar el valor actual de a en la expresión en la que esta variable reside, después incrementar a en 1.
--	predecremento	--b	Decrementar b en 1, después utilizar el nuevo valor de b en la expresión en que esta variable reside.
--	postdecremento	b--	Usar el valor actual de b en la expresión en la que esta variable reside, después decrementar b en 1.

Ejemplo:

```
int c = 5;
cout << c << endl; // imprime 5
cout << c++ << endl; // imprime 5 y después postincrementa
cout << c << endl; // imprime 6

c = 5; // asigna 5 a c
cout << c << endl; // imprime 5
cout << ++c << endl; // preincrementa y después imprime 6
cout << c << endl; // imprime 6
```

Fuente: Como programar en C++, Deitel & Deitel, 9 Edi

Operadores lógicos

Al igual que los operadores relacionales, los lógicos también son binarios, actúan sobre dos operandos (expresiones), y dan como resultado, un valor lógico (***true*** o ***false***).

C++ proporciona los operadores lógicos:

- **&&** (AND lógico) Devuelve ***true*** solo cuando ambas expresiones sean ***true,*** en caso contrario ***false***
- **||** (OR lógico) Devuelve ***true*** cuando cualquiera de las expresiones sea ***true,*** en caso contrario ***false***
- **!** (NOT o negación lógica) Cambia el valor de una expresión. Es decir si la expresión es ***true*** al negarla pasa a ser ***false***, o si la expresión es ***false*** al negarla queda como ***true***

A continuación una tabla de verdad para dos variables **X** y **Y**, que muestra el resultado de cada operador lógico.

X	Y	X && Y	X || Y	!X	!(!X)	! (X && Y)
T	T	T	T	F	T	F
T	F	F	T	F	T	T
F	T	F	T	T	F	T
F	F	F	F	T	F	T

Ejemplos de Operadores lógicos

Ejemplo 1:

(true == false) && (false != true)

false && true

false

Ejemplo 2:

(5 >= 10) || (5 != 10)

false || true

true

Ejemplo 3:

(8 % 2 == 0) && (19 >= 14)

(0 == 0) && true

true && true

true

Ejemplo 4:

(1 < 8) && (25 >= 26) || !(7 == -7)

true && false || !(false)

true && false || true

false || true

true