Abbreviated Precedence Chart for C++ Operators*

Kenneth Leroy Busbee

Abstract

An abbreviated precedence chart for C++ operators typically used in a modular structured programming fundamentals course.

An **operator** is a language-specific syntactical token (one or more symbols) that causes an action to be taken on one or more operands. The following item provides an abbreviated list of those C++ operators that are typically taught in a programming fundamentals course that teaches modular structured programming concepts.

The first column shows the precedence (the higher precedence is 1 or it goes first) and operators that have the same precedence also have the same associativity (the associativity is only listed once for the group of operators). Decrement is two minus signs, but some word processing software programs might have problems printing two minus signs and convert it to a double dash. Insertion (two < signs) and extraction (two > signs) might also have printing problems. These printing problems are noted in the comments with **emphasized** text.

PR	OPERATOR NAME	SYMBOL(S)	COMMENTS	ASSOICIATIVITY	CONNEXIONS MODULE
1	function call	()		Left to Right	m19145[75]
1	index	[]	aka array index		m21316[76]
2	class member	.	a period	Right to Left	m20796[77]
2	postfix increment	++	unary		m20499[78]
2	postfix decrement	–	unary, **two minus signs**		m20499[79]

continued on next page

*Version 1.13: Jan 14, 2010 5:50 am -0600
†http://creativecommons.org/licenses/by/2.0/

3	indirection	*	unary, aka dereference	Right to Left	m22152[80]
3	address	&	unary		m22148[81]
3	unary positive	+	unary, aka plus		m20501[82]
3	unary negative	-	unary, aka minus		m20501[83]
3	prefix increment	++	unary		m20499[84]
3	prefix decrement	–	unary, **two minus signs**		m20499[85]
3	cast	(type)	unary		m18744[86]
3	sizeof	sizeof (type)	unary		m18736[87]
3	logical NOT	!	unary		m19847[88]
4	multiply	*		Left to Right	m18706[89]
4	divide	/			m18706[90]
4	modulus	%	remainder		m18706[91]
5	add	+		Left to Right	m18706[92]
5	subtract	-			m18706[93]
6	insertion	≪	writing, **two less than signs**	Left to Right	m18835[94]
6	extraction	≫	reading, **two greater than signs**		m18835[95]
7	less than	<		Left to Right	m19549[96]
7	greater than	>			m19549[97]
7	less than or equal to	<=			m19549[98]
7	greater than or equal to	>=			m19549[99]
8	equality	==	equal to	Left to Right	m19549[100]

continued on next page

8	inequality	!=	not equal to		m19549[101]
9	logical AND	&&		Left to Right	m19847[102]
10	logical OR	||		Left to Right	m19847[103]
11	conditional	? :	trinary	Left to Right	m20811[104]
12	assignment	=		Right to Left	m18725[105]
12	addition assignment	+=			m18743[106]
12	subtraction assignment	-=			m18743[107]
12	multiplication assignment	*=			m18743[108]
12	division assignment	/=			m18743[109]
12	modulus assignment	%=			m18743[110]
13	sequence or comma	,		Left to Right	m18690[111]

Table 1

[75] "Program Control Functions" <http://cnx.org/content/m19145/latest/>
[76] "Array Index Operator" <http://cnx.org/content/m21316/latest/>
[77] "String Class within C++" <http://cnx.org/content/m20796/latest/>
[78] "Increment and Decrement Operators" <http://cnx.org/content/m20499/latest/>
[79] "Increment and Decrement Operators" <http://cnx.org/content/m20499/latest/>
[80] "Indirection Operator" <http://cnx.org/content/m22152/latest/>
[81] "Address Operator" <http://cnx.org/content/m22148/latest/>
[82] "Unary Positive and Negative Operators" <http://cnx.org/content/m20501/latest/>
[83] "Unary Positive and Negative Operators" <http://cnx.org/content/m20501/latest/>
[84] "Increment and Decrement Operators" <http://cnx.org/content/m20499/latest/>
[85] "Increment and Decrement Operators" <http://cnx.org/content/m20499/latest/>
[86] "Data Type Conversions" <http://cnx.org/content/m18744/latest/>
[87] "Sizeof Operator" <http://cnx.org/content/m18736/latest/>
[88] "Logical Operators" <http://cnx.org/content/m19847/latest/>
[89] "Arithmetic Operators" <http://cnx.org/content/m18706/latest/>
[90] "Arithmetic Operators" <http://cnx.org/content/m18706/latest/>
[91] "Arithmetic Operators" <http://cnx.org/content/m18706/latest/>
[92] "Arithmetic Operators" <http://cnx.org/content/m18706/latest/>
[93] "Arithmetic Operators" <http://cnx.org/content/m18706/latest/>
[94] "Standard Input and Output" <http://cnx.org/content/m18835/latest/>
[95] "Standard Input and Output" <http://cnx.org/content/m18835/latest/>
[96] "Relational Operators" <http://cnx.org/content/m19549/latest/>
[97] "Relational Operators" <http://cnx.org/content/m19549/latest/>
[98] "Relational Operators" <http://cnx.org/content/m19549/latest/>
[99] "Relational Operators" <http://cnx.org/content/m19549/latest/>
[100] "Relational Operators" <http://cnx.org/content/m19549/latest/>
[101] "Relational Operators" <http://cnx.org/content/m19549/latest/>
[102] "Logical Operators" <http://cnx.org/content/m19847/latest/>
[103] "Logical Operators" <http://cnx.org/content/m19847/latest/>
[104] "Conditional Operator" <http://cnx.org/content/m20811/latest/>
[105] "Assignment Operator" <http://cnx.org/content/m18725/latest/>
[106] "Arithmetic Assignment Operators" <http://cnx.org/content/m18743/latest/>
[107] "Arithmetic Assignment Operators" <http://cnx.org/content/m18743/latest/>
[108] "Arithmetic Assignment Operators" <http://cnx.org/content/m18743/latest/>
[109] "Arithmetic Assignment Operators" <http://cnx.org/content/m18743/latest/>
[110] "Arithmetic Assignment Operators" <http://cnx.org/content/m18743/latest/>
[111] "Sequence Operator" <http://cnx.org/content/m18690/latest/>