Rank	Operator in C++	Description	Result	Associativity
A	()	Grouping		N/A
A	::	Scope resolution operator, unary (global)		N/A
A	::	Scope resolution operator, binary		L-R
B1	()	Function call	rexp	L-R
B2	[]	Subscript	lexp	L-R
B3	.	Structure member	lexp	L-R
B4	->	Structure pointer member	lexp	L-R
B5	++	Postfix increment	rexp	L-R
B6	--	Postfix decrement	rexp	L-R
C1	!	Logical negate	rexp	R-L
C2	~	One's complement	rexp	R-L
C3	+	Unary plus	rexp	R-L
C4	-	Unary minus	rexp	R-L
C5	++	Prefix increment	**lexp**	R-L
C6	--	Prefix decrement	**lexp**	R-L
C7	*	Indirection (dereference)	lexp	R-L
C8	&	Address of	rexp	R-L
C9	sizeof	Size in bytes	rexp	R-L
D	(type)	Type conversion (cast)	rexp	R-L
E1	*	Multiplication	rexp	L-R
E2	/	Division	rexp	L-R
E3	%	Integer remainder (modulo)	rexp	L-R
F1	+	Addition	rexp	L-R
F2	-	Subtraction	rexp	L-R
G1	<<	Left shift	rexp	L-R
G2	>>	Right shift	rexp	L-R
H1	>	Greater than	rexp	L-R
H2	>=	Greater than or equal	rexp	L-R
H3	<	Less than	rexp	L-R
H4	<=	Less than or equal	rexp	L-R
I1	==	Equal to	rexp	L-R
I2	!=	Not equal to	rexp	L-R
J	&	Bitwise AND	rexp	L-R
K	^	Bitwise exclusive OR	rexp	L-R
L	|	Bitwise inclusive OR	rexp	L-R
M	&&	Logical AND	rexp	L-R
N	||	Logical OR	rexp	L-R
O	?:	Conditional	**lexp**	N/A
P1	=	Assignment	**lexp**	R-L
P2	+=	Add to	**lexp**	R-L
P3	-=	Subtract from	**lexp**	R-L
P4	*=	Multiply by	**lexp**	R-L
P5	/=	Divide by	**lexp**	R-L
P6	%=	Modulo by	**lexp**	R-L
P7	<<=	Shift left by	**lexp**	R-L
P8	>>=	Shift right by	**lexp**	R-L
P9	&=	AND with	**lexp**	R-L
P10	^=	Exclusive OR with	**lexp**	R-L
P11	|=	Inclusive OR with	**lexp**	R-L
Q	,	Comma	rexp	L-R

Note: All operators within a section (between horizontal lines) have the same precedence and the associativity must be applied.

Some non-printing control characters

0	NUL
7	Bell
8	Backspace
9	Tab
10	Line feed
13	Carriage return
26	End of file (Ctrl-Z)
27	[Esc] (Escape key)

ASCII characters (only 32-127 are standard)

32		64	@	96	`	128	Ç	160	á	192	└	224	α
33	!	65	A	97	a	129	ü	161	í	193	┴	225	ß
34	"	66	B	98	b	130	é	162	ó	194	┬	226	Γ
35	#	67	C	99	c	131	â	163	ú	195	├	227	π
36	$	68	D	100	d	132	ä	164	ñ	196	─	228	Σ
37	%	69	E	101	e	133	à	165	Ñ	197	┼	229	σ
38	&	70	F	102	f	134	å	166	ª	198	╞	230	µ
39	'	71	G	103	g	135	ç	167	º	199	╟	231	τ
40	(	72	H	104	h	136	ê	168	¿	200	╚	232	Φ
41	)	73	I	105	i	137	ë	169	⌐	201	╔	233	Θ
42	*	74	J	106	j	138	è	170	¬	202	╩	234	Ω
43	+	75	K	107	k	139	ï	171	½	203	╦	235	δ
44	,	76	L	108	l	140	î	172	¼	204	╠	236	∞
45	-	77	M	109	m	141	ì	173	¡	205	═	237	φ
46	.	78	N	110	n	142	Ä	174	«	206	╬	238	ε
47	/	79	O	111	o	143	Å	175	»	207	╧	239	∩
48	0	80	P	112	p	144	É	176	░	208	╨	240	≡
49	1	81	Q	113	q	145	æ	177	▒	209	╤	241	±
50	2	82	R	114	r	146	Æ	178	▓	210	╥	242	≥
51	3	83	S	115	s	147	ô	179	│	211	╙	243	≤
52	4	84	T	116	t	148	ö	180	┤	212	╘	244	⌠
53	5	85	U	117	u	149	ò	181	╡	213	╒	245	⌡
54	6	86	V	118	v	150	û	182	╢	214	╓	246	÷
55	7	87	W	119	w	151	ù	183	╖	215	╫	247	≈
56	8	88	X	120	x	152	ÿ	184	╕	216	╪	248	°
57	9	89	Y	121	y	153	Ö	185	╣	217	┘	249	∙
58	:	90	Z	122	z	154	Ü	186	║	218	┌	250	·
59	;	91	[	123	{	155	¢	187	╗	219	█	251	√
60	<	92	\	124	|	156	£	188	╝	220	▄	252	ⁿ
61	=	93	]	125	}	157	¥	189	╜	221	▌	253	²
62	>	94	^	126	~	158	₧	190	╛	222	▐	254	■
63	?	95	_	127	⌂	159	ƒ	191	┐	223	▀	255	

Common **printf** formatting codes

```
%c - characters
%s - strings (NUL-terminated C strings)
%d, %i - integers
%f - floating point
%g - floating point (minimum digits)
%e - scientific notation
%p - pointers (displays in hex)
%x - hexadecimal integers (Use %X for uppercase)
%o - octal integers
%u - unsigned integers
%ld, %li - long integers
%lu - unsigned long integers
%hd, %hi - short integers
%hu - unsigned short integers
```