

Glossary of C/C++ Symbols

Symbol(s)	Pronunciation	Description/Usage/Context
{	left brace	Starts a new block of code, where you can declare local variables
}	right brace	Ends the current code block; all local variables disappear -- "go out of scope"
{ }	braces	Enclose a block of code

Symbol(s)	Pronunciation	Description/Usage/Context
(left parenthesis	Starts an if or while condition; used to force order of evaluation in an expression; starts a for loop; starts a function argument list; starts a function call argument list
)	right parenthesis	Ends an if or while or for statement; used with a left parenthesis to force order of evaluation in an expression; ends a function parameter list; ends a function call parameter list

Symbol(s)	Pronunciation	Description/Usage/Context
()	Parentheses	if/while/for statements; used in mathematical and logical expressions to force order of evaluation; used for function declarations, function definitions, and function calls
[Left square bracket	To declare the size of an array; to indicate an array element index
]	Right square bracket	"
[]	Square brackets	"
<	Less than	
>	Greater than	
<=	Less than or equal to	

Symbol(s)	Pronunciation	Description/Usage/Context
>=	Greater than or equal to	
=	Equals	Assignment operator -- make the left side equal to the whatever the right side is evaluated as
==	Double equals Is equal to	Equality operator; evaluates to true if left side has same value as the right side
!=	Is not equal to	Evaluates to false if left side has same value as the right side
!	Not	Inverts result of boolean expression: true goes to false, false goes to true

Symbol(s)	Pronunciation	Description/Usage/Context
<<	Insertion operator	Used with cout or any other ostream object; inserts data into the stream
<<	Left shift operator	Shifts bits in the value on the left side to the left by the number of places indicated on the right side
>>	Extraction operator	Used with cin or any other istream object; extracts data from the stream
>>	Right shift operator	Shifts bits in the value on the left side to the right the number of places indicated on the right side

Symbol(s)	Pronunciation	Description/Usage/Context
*	Times	Multiplication when used in an arithmetic expression
*	Star, asterisk	Used to dereference a pointer variable, to get its value
#	Pound	Used for preprocessor directives
&	Ampersand, and	Bitwise AND operator: ANDs each bit of the left side with each bit of the right side
&	Ampersand, and	Address-of operator; returns the address (as a pointer) of the variable to its right
&	Ampersand, and	Used to declare reference variables or reference arguments in a function declaration

Symbol(s)	Pronunciation	Description/Usage/Context
&&	Double ampersand, double and	Logical AND operator: ANDs boolean value of left side with boolean value of right side
	Pipe, or	Bitwise OR operator: ORs each bit of the left side with each bit of the right side
	Double pipe, double or	Logical OR operator: ORs boolean value of left side with boolean value of right side
^	Caret	Bitwise XOR (exclusive OR) operator: XORs each bit of the left side with each bit of the right side

Symbol(s)	Pronunciation	Description/Usage/Context
~	Tilde, squiggly	Bitwise NOT (negation) operator: inverts each bit of the value to its right; 1's go to 0's, 0's to 1's
!	Not, bang	Logical NOT (negation) operator: inverts result of boolean expression to its right; true goes to false, false goes to true
:	Colon	Used for case labels; used in the conditional operator; used in class definitions; used for bitwise structure fields
;	Semicolon	Use to end statements; a "no-op" (i.e., do nothing) in a function or loop body

Symbol(s)	Pronunciation	Description/Usage/Context
.	Point	Decimal point
.	Dot	Direct access operator: used to access one member of a structure; used to call methods on an object
->	Arrow	Indirect access operator: used to access one member of a structure through a pointer; used to call methods via a pointer to an object
\	Backslash, escape	Escape a special character inside a string; to define a character constant; used to break up a long line across two or more lines
/	Slash	Division operator

Symbol(s)	Pronunciation	Description/Usage/Context
+	Plus	Addition operator; used for concatenation in some string classes
-	Minus	Subtraction operator; when used with a single value, negation operator
/* */	C style comment	Used in C or C++ programs; may span across multiple lines
//	C++ style comment	Used in C++ programs only; single line only
++	Plus plus	Add one to a variable; move a pointer to the next address
--	Minus minus	Subtract one from a variable; move a pointer to the previous address

Symbol(s)	Pronunciation	Description/Usage/Context
+ =	Plus equals	Add what is on the right side to a variable on the left side
- =	Minus equals	Subtract what is on the right side from a variable on the left side
* =	Times equals	Multiply the variable on the left side by whatever is on the right side
/=	Divide equals	Divide the variable on the left side by whatever is on the right side
%	Mod	Modulus operator: returns the remainder of dividing the left side by the right side (integers only)

Symbol(s)	Pronunciation	Description/Usage/Context
%	Percent	Used to begin a placeholder in a printf, fprintf, scanf, fscanf, sprintf, or sscanf format string
%%	Percent percent	To get a real percent sign in a printf, fprintf, scanf, fscanf, sprintf, or sscanf format string
%=	Mod equals	To replace the variable on the left with the remainder of dividing its value by whatever is on the right
<<=	Left shift equals	Replace the variable on the left with the result of shifting its bits to the left as many times as specified on the right side

Symbol(s)	Pronunciation	Description/Usage/Context
>>=	Right shift equals	Replace the variable on the left with the result of shifting its bits to the right as many times as specified on the right side
&=	AND equals	Replace the variable on the left with the result of ANDing its bits with whatever is on the right side
=	OR equals	Replace the variable on the left with the result of ORing its bits with whatever is on the right side

Symbol(s)	Pronunciation	Description/Usage/Context
<code>^=</code>	XOR equals	Replace the variable on the left with the result of XORing its bits with whatever is on the right side
<code>"</code>	Double quote	To begin and end a string constant/string literal
<code>'</code>	Single quote	To begin and end a character constant/character literal
<code>_</code>	Underscore	For spacing in variable, constant, and function names
<code>?</code>	Question mark	Used for the conditional operator

Symbol(s)	Pronunciation	Description/Usage/Context
,	Comma	To separate multiple expressions in a single statement; to declare several variables in a single statement; to separate lists of parameters/arguments in functions and function calls