1 Variables

The table that follows shows the most common types used in C++

OPERATOR	**DESCRIPTION**	**USAGE**

=	Binary operator to assign the value on the right to the expression on the left	int i;
		i =3;

!	Unary operator to complement the true/false (non-0/0) status of an expression.	bool b = !true;
		bool b2 = !b;

+	Binary operator for addition	int $i = 3 + 2$; int $j = i + 5$;
$-$ $*$ $/$	Binary operators for subtraction, multiplication, and division.	int $i = 5 - 1$; int $j = 5 * 2$; int $k = j/i$;
%	Binary operator for remainder of a division operation.	int remainder = 5%2;

++	Unary operator to increment an expression by 1. If the operator occurs after the expression or *post-increment*, the result of the expression is the unincremented value. If the operator occurs before the expression or *pre-increment*, the result of the expression is the new value	i++;
		++i;

$--$	Unary operator to decrement an expression by 1.	i$--$; $--$i;
$+ =$	Shorthand syntax for $i = i + j$	i + = j;
$- =$ $* =$ $/ =$ $\% =$	Shorthand syntax	$i = i - j$; $i = i * j$; $i = i/j$; $i = i\%j$;
& &=	Takes the raw bits of one expression and performs a bitwise ”AND” with the other expression	$i = j\&k$; $j\& = k$;
$\vert$ $\vert =$	Takes the raw bits of one expression and performs a bitwise ”OR” with the other expression	$i = j\vert k$; $j\vert = k$;
>> << <<= >>=	Takes the raw bits of an expression and ”shifts” each bit left (<<) or right (>>) the specified number of places.	$i = i << 1$; $i = i >> 4$; $i <<= 1$; $i >>= 4$;
ˆ ˆ=	Performs a bitwise ”exclusive or,” also called ”XOR” operation, on the two expressions.	$i = i\hat{}j$; $i\hat{} = j$;