



- Arithmetic Operator

- Assume variable A holds 10 and variable B holds 20
- A = 10, B = 20;

Operator	Description	Example
+	Adds two operands	A + B will give 30
-	Subtracts second operand from the first	A - B will give -10
*	Multiplies both operands	A * B will give 200
/	Divides numerator by de-numerator	B / A will give 2
%	Modulus Operator and remainder of after an integer division	B % A will give 0
++	Increment operator  , increases integer value by one	A++ will give 11
--	Decrement operator  , decreases integer value by one	A-- will give 9

- Relational Operators

- Assume variable A holds 10 and variable B holds 20
- A = 10, B = 20;

Operator	Description	Example
==	Checks if the values of two operands are equal or not, if yes then condition becomes true.	(A == B) is not true.
!=	Checks if the values of two operands are equal or not, if values are not equal then condition becomes true.	(A != B) is true.
>	Checks if the value of left operand is greater than the value of right operand, if yes then condition becomes true.	(A > B) is not true.
<	Checks if the value of left operand is less than the value of right operand, if yes then condition becomes true.	(A < B) is true.
>=	Checks if the value of left operand is greater than or equal to the value of right operand, if yes then condition becomes true.	(A >= B) is not true.
>=	Checks if the value of left operand is greater than or equal to the value of right operand, if yes then condition becomes true.	(A >= B) is not true.

- Logical Operators

- Assume A holds 1 and B holds 0
- A = 1, B = 0;

Operator	Description	Example
&&	Called Logical AND operator. If both the operands are non-zero, then condition becomes true.	(A && B) is false.
	Called Logical OR Operator. If any of the two operands is non-zero, then condition becomes true.	(A B) is true.
!	Called Logical NOT Operator. Use to reverses the logical state of its operand. If a condition is true, then Logical NOT operator will make false.	!(A && B) is true.

- Bitwise Operators

- Assume A holds 1 and B holds 0
- A = 1, B = 0;

p	q	p & q	p q	p ^ q
0	0	0	0	0
0	1	0	1	1
1	1	1	1	0
1	0	0	1	1

Assume if A = 60; and B = 13; now in binary format they will be as follows –

A = 0011 1100

B = 0000 1101

A&B = 0000 1100

A|B = 0011 1101

A^B = 0011 0001








~A = 1100 0011

Operator	Description	Example
&	Binary AND Operator copies a bit to the result if it exists in both operands.	(A & B) will give 12 which is 0000 1100
	Binary OR Operator copies a bit if it exists in either operand.	(A B) will give 61 which is 0011 1101
^	Binary XOR Operator copies the bit if it is set in one operand but not both.	(A ^ B) will give 49 which is 0011 0001
~	Binary Ones Complement Operator is unary and has the effect of 'flipping' bits.	(~A) will give -61 which is 1100 0011 in 2's complement form due to a signed binary number.
<<	Binary Left Shift Operator. The left operands value is moved left by the number of bits specified by the right operand.	A << 2 will give 240 which is 1111 0000
>>	Binary Right Shift Operator. The left operands value is moved right by the number of bits specified by the right operand.	A >> 2 will give 15 which is 0000 1111

- Assignment Operators

Operator	Description	Example
=	Simple assignment operator, Assigns values from right side operands to left side operand.	$C = A + B$ will assign value of $A + B$ into C
+=	Add AND assignment operator, It adds right operand to the left operand and assign the result to left operand.	$C += A$ is equivalent to $C = C + A$
-=	Subtract AND assignment operator, It subtracts right operand from the left operand and assign the result to left operand.	$C -= A$ is equivalent to $C = C - A$
*=	Multiply AND assignment operator, It multiplies right operand with the left operand and assign the result to left operand.	$C *= A$ is equivalent to $C = C * A$
/=	Divide AND assignment operator, It divides left operand with the right operand and assign the result to left operand.	$C /= A$ is equivalent to $C = C / A$
%=	Modulus AND assignment operator, It takes modulus using two operands and assign the result to left operand.	$C \% = A$ is equivalent to $C = C \% A$
<<=	Left shift AND assignment operator.	$C <<= 2$ is same as $C = C << 2$
>>=	Right shift AND assignment operator.	$C >>= 2$ is same as $C = C >> 2$
&=	Bitwise AND assignment operator.	$C \&= 2$ is same as $C = C \& 2$
^=	Bitwise exclusive OR and assignment operator.	$C \wedge= 2$ is same as $C = C \wedge 2$
=	Bitwise inclusive OR and assignment operator.	$C = 2$ is same as $C = C 2$

- Misc Operators

Sr.No	Operator & Description
1	sizeof sizeof operator  returns the size of a variable. For example, sizeof(a), where 'a' is integer, and will return 4.
2	Condition ? X : Y Conditional operator (?)  . If Condition is true then it returns value of X otherwise returns value of Y.
3	, Comma operator  causes a sequence of operations to be performed. The value of the entire comma expression is the value of the last expression of the comma-separated list.
4	. (dot) and -> (arrow) Member operators  are used to reference individual members of classes, structures, and unions.
5	Cast Casting operators  convert one data type to another. For example, int(2.2000) would return 2.
6	& Pointer operator &  returns the address of a variable. For example &a; will give actual address of the variable.
7	* Pointer operator *  is pointer to a variable. For example *var; will pointer to a variable var.