ITI

Types of operators in C

C Programming Language

Prepared by:
Eng. Ashraf Mahmoud Abdelzaher

Under the supervision of:
Eng. Ayman

Types of operators in C

There are different types of operators in C programming language:

1. Arithmetic Operator.
2. Increment/Decrement Operator.
3. Assignment Operator.
4. Relational Operator.
5. Logical Operator.
6. Bitwise Operator.
7. Misc Operator.

Arithmetic Operator

Arithmetic Operators are the operators which are used to perform mathematical calculations like addition (+), subtraction (-), multiplication (*), division (/), and modulus (%). It performs all the operations on numerical values (constants and variables).

Operator	Description	Example
+(Addition)	It adds two operands	A + B = 9
-(Subtraction)	It subtracts second operand from the first	A – B = 3
*(Multiplication)	It multiplies both operands	A * B = 20
/(Division)	It is responsible for dividing numerator by the denomerator	B / A = 8
%(Modulus)	This operator gives the remainder of an integer after division	B % A = 0

Increment/Decrement Operator

C programming has basically two operators which can increment ++ and decrement -- the value of a variable. It can change the value of an operand by 1. These two operators are unary operators, which means they can only operate on a single operand. For example, ++x and x++ means x=x+1 or --x and x−− means x=x-1.

There is a slight distinction between ++ or −− when written before or after any operand.

If we use the operator as a pre-fix, it adds 1 to the operand, and the result is assigned to the variable on the left. Whereas, when it is used as a post-fix, it first assigns the value to the variable on the left i.e., it first returns the original value, and then the operand is incremented by 1.

Operator	Description	Example
++	This increment operator increases the integer value by 1.	A++ = 2
--	This decrement operator decreases the integer value by 1.	A-- = 3

Assignment Operator

An assignment operator is mainly responsible for assigning a value to a variable in a program.

Operator	Description	Example
=	Used to assign the values from right side of the operands to left side of the operand.	C = A + B will assign the value of A + B to C.
+=	Adds the value of the right operand to the value of the left operand and assigns the result to the left operand.	C += A is same as C = C + A
-=	Subtracts the value of the right operand from the value of the left operand and assigns the result to the left operand.	C -= A is same as C = C - A
*=	Multiplies the value of the right operand with the value of the left operand and assigns the result to the left operand.	C *= A is same as C = C * A
/=	Divides the value of the left operand with the value of the right operand and assigns the result to the left operand.	C /= A is same as C = C / A
%=	Takes modulus using the values of the two operands and assigns the	C %= A is same as C = C % A

	result to the left operand.	
<<=	Used for left shift AND assignment operator.	C <<= 4 is same as C = C << 4
>>=	Used for right shift AND assignment operator.	C >>= 5 is same as C = C >> 5
&=	Used for bitwise AND assignment operator.	C &= 7 is same as C = C & 7
^=	Used for bitwise exclusive OR and assignment operator.	C ^= 6 is same as C = C ^ 6
|=	Used for bitwise inclusive OR and assignment operator.	C |= 9 is same as C = C | 9

Relational Operator

Relational operators are specifically used to compare two quantities or values in a program. It checks the relationship between two operands. If the given relation is true, it will return 1 and if the relation is false, then it will return 0. Relational operators are heavily used in decision-making and performing loop operations.

Note: assume that the variable A holds 10 and the variable B holds the 20.

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

	the condition becomes true.	
>=	It is used to check if the value of left operand is greater than or equal to the value of right operand. If the value of the left operand is greater than or equal to the value, then the condition becomes true.	(A >= B) is not true.
<=	It is used to check if the value of left operand is less than or equal to the value of right operand. If the value of the left operand is less than or equal to the value, then the condition becomes true.	(A <= B) is true.

Logical Operator

In the C programming language, we have three logical operators when we need to test more than one condition to make decisions. Logical operators are generally used for decision-making in C programming.
Note: assume that the variable A holds 7 and the variable B holds the 3.

Operator	Description	Example
&&	This is the AND operator in C programming language. It performs logical conjunction of two expressions. (If both expressions evaluate to True, then the result is True. If either of the expression evaluates to False, then the result is False)	((A==7) && (B>7)) equals to 0
||	It is the NOT operator. It performs a logical disjunction on two expressions. (If either or both of the expressions evaluate to True, then the result is True)	((A==7) || (B>7)) equals to 1
!	It is the Logical NOT Operator. It is used to reverse the logical state of its operand. If a condition is true, then the Logical NOT operator will make it false and vice versa.	!(A && B) is true

Bitwise Operator

Bitwise operators are the operators which work on bits and perform the bit-by-bit operation.

Note: assume that the variable A holds 50 and the variable B holds the 25.

Operator	Description	Example
&	**Binary AND Operator. It copies a bit to the result if it exists in both the operands.**	**(A & B) = 16, i.e. 00010000**
|	**Binary OR Operator. It copies a bit if and only if it exists in either operand.**	**(A | B) = 59, i.e. 00111011**
^	**Binary XOR Operator. It copies the bit only if it is set in one operand but not both.**	**(A ^ B) = 43, i.e. 00101011**
~	**Binary One's Complement Operator. It is unary and has the effect of 'flipping' bits.**	**(~A) = ~(50), i.e,. -0111101**

<<	**Binary Left Shift Operator. The value of the left operands is moved left by the number of bits specified by the right operand.**	**A << 2 = 200 i.e. 11001000**
>>	**Binary Right Shift Operator. The value of the left operands is moved right by the number of bits specified by the right operand.**	**A >> 2 = 12 i.e., 00001100**

Misc Operator

C programming language also offers a few other important operators including sizeof, comma, pointer (*), and conditional operator (?:).

Operator	Description	Example
sizeof()	The sizeof is a unary operator that returns the size of data (constants, variables, array, structure, etc).	sizeof(a), where a is integer, will return 4. sizeof(b), where b is float, will return 4. sizeof(c), where c is double, will return 8. sizeof(d), where d is integer, will return 1.
&	It returns the address of a memory location of a variable.	&a; returns the actual address of the variable. It can be any address in the memory like 4, 70,104.
*	Pointer to a variable.	*a; It points to the value of the variable.
? :	conditional operator (? : in combination) to construct conditional expressions.	If Condition is true? then value X : otherwise value Y will be returned as output.