Operators

Unit 3

INTRODUCTION

In the previous units (unit 1 and 2) you have learned about the basics of computer programming, different data types, constants, keywords and basic structure of a C program. You've learned how to declare and initialize variables and constant. In this unit you will learn about operator, Operators are special symbols that perform specific operations on one, two, or three operands, and then return a result; and finally will explore the operators available in C programming language.

Timeframe	
How long ?	We expect that this unit will take maximum 3 hours to complete.

Unit Structure	
Lesson- 1 : Operator part -1 Lesson- 2 : Operator part -2	

Lesson -1 Operator Part-1

Learning Outcomes

Outcomes

Upon completion of this lesson you will be able to

- Define Operator.
- Classify Operator.
- Explain arithmetic operator.

☑	**Keywords**	**Operator, Arithmetic, Precedence**

OPERATOR

An operator is a symbol that tells the compiler to perform specific mathematical or logical functions. Operators, functions, constants and variables are combined together to form expressions. Consider the expression A + B * 5 where, +, * are operators, A, B are variables, 5 is constant and A + B * 5 is an expression. C language is rich in built-in operators and provides the following types of operators:

Types of Operators	Description
Arithmetic operators	These are used to perform mathematical calculations like addition, subtraction, multiplication, division and modulus
Assignment operators	These are used to assign the values for the variables in C programs.
Relational operators	These operators are used to compare the value of two variables.
Logical operators	These operators are used to perform logical operations on the given two variables.
Bit wise operators	These operators are used to perform bit operations on given two variables.
Conditional (ternary) operators	Conditional operators return one value if condition is true and returns another value is condition is false.
Increment/decrement operators	These operators are used to either increase or decrease the value of the variable by one.
Special operators	&, *, sizeof( ) and ternary operators.

ARITHMETIC OPERATORS

Computer programs are widely used for mathematical calculations. In C language we have the usual arithmetic operators for addition, subtraction, multiplication and division. C also provides a special arithmetic operator which is called modulus. All these operators are binary operators which means they operate on two operands. So we need two values for addition, subtraction, multiplication, division and modulus. The following table lists down a few of the important arithmetic operators available in C programming language. Assume variable x holds 10 and variable y holds 20, then:

OPERATOR	OPERATION	DESCRIPTIONS	C EXPRESSION	EXAMPLE
+	Addition	Adds two operands	x + y	x + y = 30
-	Subtraction	Subtracts second operand from the first	x - y	x-y = 10
*****	Multiplication	Multiplies both operands	x * y	x * y = 200
/	Division	Divides numerator by de-numerator	x / y	x / y = 2
%	Modulus	provide remainder of an integer division	x % y	x % y = 0

Addition, subtraction and multiplication are same as we use in algebra. There is one thing to note in division that when we use integer division (i.e. both operands are integers) yields an integer result. This means that if, for example, you are dividing 5 by 2 (5 / 2) it will give integer result as 2 instead of actual result 2.5. Thus in integer division the result is truncated to the whole number, the fractional part (after decimal) is ignored. If we want to get the correct result, then we should use float data type. The modulus operator returns the remainder after division. This operator can only be used with integer operands. The expression x % y returns the remainder after x is divided by y. For example, the result of 5 % 2 will be 1, 23 % 5 will be 3 and 107%10 will be 7.

INCREMENT, DECREMENT OPERATORS (++, --)

Increment and decrement are common arithmetic operation. C provides two short cuts for the same.

Postfix- will return the current value and then increment/ decrement.

x++ is a short cut for x=x+1; will return the current value and then increment.

x-- is a short cut for x=x-1; will return the current value and then decrement.

Prefix:

++x is a short cut for x=x+1; will increment x and then return the value (so it will return one greater than the original value)

--x is a short cut for x=x-1; will decrement a and then return the value (so it will return one less than the original value)

PRECEDENCE OF OPERATORS

The arithmetic operators in an expression are evaluated according to their precedence. The precedence means which operator will be evaluated first and which will be evaluated after that and so on. In an expression, the parentheses ( ) are used to force the evaluation order. The operators in the parentheses ( ) are evaluated first. If there are nested parentheses then the inner most is evaluated first. The expressions are always evaluated from left to right. The operators *, / and % have the highest precedence after parentheses. These operators are evaluated before + and – operators. Thus + and – operators has the lowest precedence. It means that if there are * and + operators in an expression then first the * will be evaluated and then its result will be added to other operand. If there are * and / operators in an expression then the operator which occurs first from left will be evaluated first and then the next, except you force any operator to evaluate by putting parentheses around it. The following table explains the precedence of the arithmetic operators:

OPERATOR	OPERATIONS	PRECEDENCE (ORDER OF EVALUATION)
( )	Parentheses	Evaluated first
***, /, or %**	Multiplication, Division, Modulus	Evaluated second. If there are several, they are evaluated from left to right
+ or -	Addition, Subtraction	Evaluated last. If there are several, they are evaluated from left to right

Let’s look some examples. What is the result of 10 + 10 * 5? The answer is 60 not 100. As * has higher precedence than + so 10 * 5 is evaluated first and then the answer 50 is added to 10 and we get the result 60. The answer will be 100 if we force the addition operation to be done first by putting 10 + 10 in parentheses. Thus the same expression rewritten as (10 + 10) * 5 will give the result 100. Note that how the parentheses affect the evaluation of an expression. Similarly the expression 5 * 3 + 6 / 3 gives the answer 17, and not 7. The evaluation of this expression can be clarified by writing it with the use of parentheses as (5 * 3) + (6 /3) which gives 15 + 2 = 17. Thus you should be careful while writing arithmetic expressions.

Activity

1. Suppose a, b and c are integer variables that have been assigned the values a = 8, b = 3 and c = -5. Determine the value of each of the following arithmetic expressions.
   a) a +b + c
   b) 2 * b + 3 * (a-c)
   c) a / b
   d) a % b
   e) a * b / c
   f) a * (b / c)
   g) (a * c) % b
   h) a * (c % b).
2. Is 2 + 5 * 2 equal to (2 + 5) * 2?

3. Does 7 % 2 produce the same result as 4 % 3?

Lesson-2 Operator Part-2

Learning Outcomes

Outcomes

Upon completion of this lesson you will be able to

- Define relational operator.
- Use logical Operator.
- Explain assignment operator.

☑	**Keywords**	**Relational, Logical, Bitwise**

RELATIONAL OPERATOR

A relational operator takes a pair of expressions as operands and returns a logical value, true or false. There are six common relational operators: ==, !=, >, <   ≥ , ≤ , . The following table shows all the relational operators supported by C. Assume variable x holds 10 and variable y holds 20, then:

OPERATOR	DESCRIPTION	EXAMPLE
= =	Checks if the values of two operands are equal or not. If yes, then the condition becomes true.	(x == y) is not true.
!=	Checks if the values of two operands are equal or not. If the values are not equal, then the condition becomes true.	(x ! = y) is true.
>	Checks if the value of left operand is greater than the value of right operand. If yes, then the condition becomes true.	(x > y) is not true.
<	Checks if the value of left operand is less than the value of right operand. If yes, then the condition becomes true.	(x < y) is true.
>=	Checks if the value of left operand is greater than or equal to the value of right operand. If yes, then the condition becomes true.	(x >= y) is not true.
<=	Checks if the value of left operand is less than or equal to the value of right operand. If yes, then the condition becomes true.	(x <= y) is true.

LOGICAL OPERATOR

Logical operators are very important in any programming language and they help us take decisions based on certain conditions.

OPERATOR	OPERATIONS	DESCRIPTIONS	EXAMPLE
&&	Logical AND	If both the operands are true, then condition becomes true.	(x>5)&&(y<5)
||	Logical OR	If any of the two operands is true, then condition becomes true.	(x>=10)||(y>=10)
!	Logical NOT	Use to reverses the logical state of its operand. If a condition is true then Logical NOT operator will make false.	!((x>5)&&(y<5))

ASSIGNMENT OPERATOR

Operator	Description	Example
=	Simple assignment operator. Assigns values from right side operands to left side operand.	C = A + B will assign the value of A + B to C

In C programs, values for the variables are assigned using assignment operators. For example, if the value "10" is to be assigned for the variable "sum", it can be assigned as "sum = 10;" In C language equal-to-sign (=) is used as assignment operator. Do not confuse the algebraic equal-to with the assignment operator. In Algebra X = 2 means the value of X is 2, whereas in C language X = 2 (where X is a variable name) means take the value 2 and put it in the memory location labeled as X, afterwards you can assign some other value to X, for example you can write X = 10, that means now the memory location X contains the value 10 and the previous value 2 is no more there. In C language the statement X = X + 1 means that add 1 to the value of X and then store the result in X variable. If the value of X is 10 then after the execution of this statement the value of X becomes 11. This is a common practice for incrementing the value of the variable by 'one in C language. Similarly you can use the statement X = X - 1 for decrementing the value of the variable by one. Remember that assignment operator must have a variable name on left hand side unlike algebra in which you can use expression on both sides of equal sign (=). For example, in algebra, X +5 = Y + 7 is correct but incorrect in C language. The compiler will not understand it and will give error.

C provides compound assignment operators that can be used instead.

```
x+=1 /*is the same as x=x+1*/
x-=1 /*is the same as x=x-1*/
x*=10 /*is the same as x=x*10 */
x/=2 /* is the same as x=x/2
x%=2 /*is the same as x=x%2
```

Note that the "==" equality operator is different from the "=", assignment operator.

TERNARY OPERATOR (?:)

Conditional operators return one value if condition is true and returns another value is condition is false. This operator is also called as ternary operator.

Syntax : (Condition? true_value: false_value);
Example : (A > 100 ? 0 : 1);

In above example, if A is greater than 100, 0 is returned else 1 is returned. This is equal to if else conditional statements.

One more example

```
if(a > b)
        result = x;
else
        result = y;
is equivalent to
result = a > b ? x : y;
```

Study skills

1. What is the output of this C code?

```
#include <stdio.h>
int main()
{
    int a = 1, b = 1, c;
    c = a++ + b;
    printf("%d, %d", a, b);
}
```

a) a = 1, b = 1 b) a = 2, b = 1
c) a = 1, b = 2 d) a = 2, b = 2

Correct answer is: b

2. What is the output of this C code?

```
#include <stdio.h>
int main()
{
    int a = 10, b = 10;
    if (a = 5)
    b--;
    printf("%d, %d", a, b--);
}
```

a) a = 10, b = 9 b) a = 10, b = 8
c) a = 5, b = 9 d) a = 5, b = 8

Summary

Summary

In this lesson

- We have understood about various types of C operators and operands.
- Also understood operator precedence.

Assessment

Assessment

Fill in the blanks

3. In C logical AND represents as…………
4. …….. operator is called “modulus operator” in C

Multiple Choice Questions (MCQ)

1. Which of the following is not an arithmetic operation?
   e) a *= 10;
   f) a /= 10;
   g) a != 10;
   h) a %= 10;
2. The operator + in a+=4 means
   a) a = a + 4
   b) a + 4 = a
   c) a = 4
   d) a = 4 + 4

3. The most commonly used assignment operator is
   a) =
   b) ==
   c) >=
   d) :=

Exercises

1. State C operators and their classification.
2. What is an operand? What is the relationship between operators and operands?
3. Write the difference between = and == in C programming.
4. Describe the arithmetic, relational, logical, assignment, increment, decrement and conditional operators with suitable example.
5. Describe the precedence of arithmetic operators.
6. Write programs using operators.
7. Describe the four relational operators include in C. With what type of operands can they be used?