PROGRAMMING

FOR PROBLEM SOLVING

{ C LANGUAGE }

Module :- 2

Start

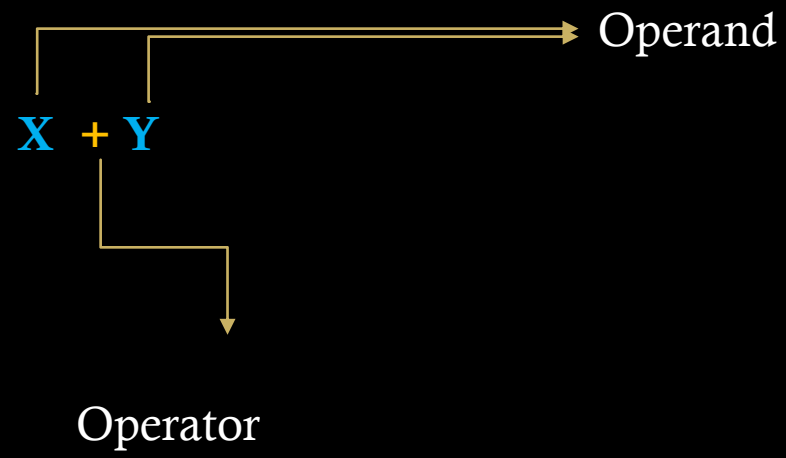
Module 2 :- Operators

- Arithmetic expressions / arithmetic operators
- relational operators
- logical operators
- bitwise operators
- precedence

-:Expression and operators :-

- **Operator :-** It is a special symbol which is used to perform logical and mathematical operation on data or variable

- **Operand :-** It is a data or variable on which the operation is to be performed



Types of operator

- C language includes a large number of operators which as :-

1. Arithmetic operator
2. Assignment operator
3. Increment and decrement operator
4. Relational operator
5. Logical operator
6. Conditional operator
7. Bitwise operator
8. Other operator
   - comma operator
   - sizeof operator

Arithmetic operator :-

- An arithmetic operator performs mathematical operation such as addition , multiplication , division , subtraction etc.. On numerical value (constant & variable)

Operator	Meaning
+	addition or unary plus
-	subtraction or unary minus
*	multiplication
/	division
%	modulo (remainder)

Note :-

% operator works only with integer

Example on arithmetic operator

1. Addition of two number
2. subtraction of two number
3. multiplication of two number
4. division of two number
5. remainder of two number

6. Area of rectangle
7. Area of square
8. Area of circle
9. Area of triangle

10. Input marks of five subject like math, physics, chemistry, hindi, English and find average
11. Simple interest

12. Swapping using third variable method 1
13. Swapping using third variable method 2
14. Swapping without using third variable

Assignment operator :-

- An arithmetic operator is used for assigning a value to a variable

Operator	example	same as
=	x=y	x=y
+=	x+=y	x=x+y
-=	x-=y	x=x-y
*=	x*=y	x=x*y
/=	x/=y	x=x/y
%=	x%=y	x=x%y

Example on assignment operator

Using assignment operator/short hand operator perform

1. Addition of two number
2. subtraction of two number
3. multiplication of two number
4. division of two number
5. remainder of two number

Increment and decrement operator

- It uses operator (++) or (- -) , ++ means increment value by one and - - means decrement value by one.
- This operator used with only variable
- These are two types of increment and decrement operator

1. Prefix
2. Postfix

- **Prefix :-** here first the value of variable is incremented or decremented by one then the new value is assign to the variable

Eg.

```
y = ++x          y = - -x
x = x+1          x = x-1
y = x            y = x
```

- **Postfix :-** here first the value of variable is used in operation and then increment or decrement operation performed

Eg.

```
y = x++          y = x- -
y = x            y = x
x = x + 1        x = x - 1
```

Operator	name	function	example
+ +	increment	it increments the value by 1	++x
- -	decrement	it decrement the value by 1	- - x

Example on increment and decrement operator

Go to code editor software

Relational operator

- A relational operator checks the relationship between two operands if the relation is ture it returns 1 if the relation is false it return value 0
- It used in decision making and loop in c programming

Operator	meaning	example (a=5 ,b =3)
==	equal to	a==b is evaluated to 0
>	greater than	a>b is evaluated to 1
<	less then	a<b is evaluated to 0
!=	not equal to	a!=b is evaluated to 1
>=	greater than or equal to	a>=b is evaluated to 1
<=	less then or equal to	a<=b is evaluated to 0

Example on relational operator

Go to code editor software

Logical operator :-

- An expression containing logical operation return either **0** or **1** depending whether expression result true or false
- This operator commonly used in decision making programming

Operator	meaning	example
&&	logical AND	if (x>y && x>z)
||	logical OR	if (x>y || x>z)
!	Logical NOT	if (!(x>y))

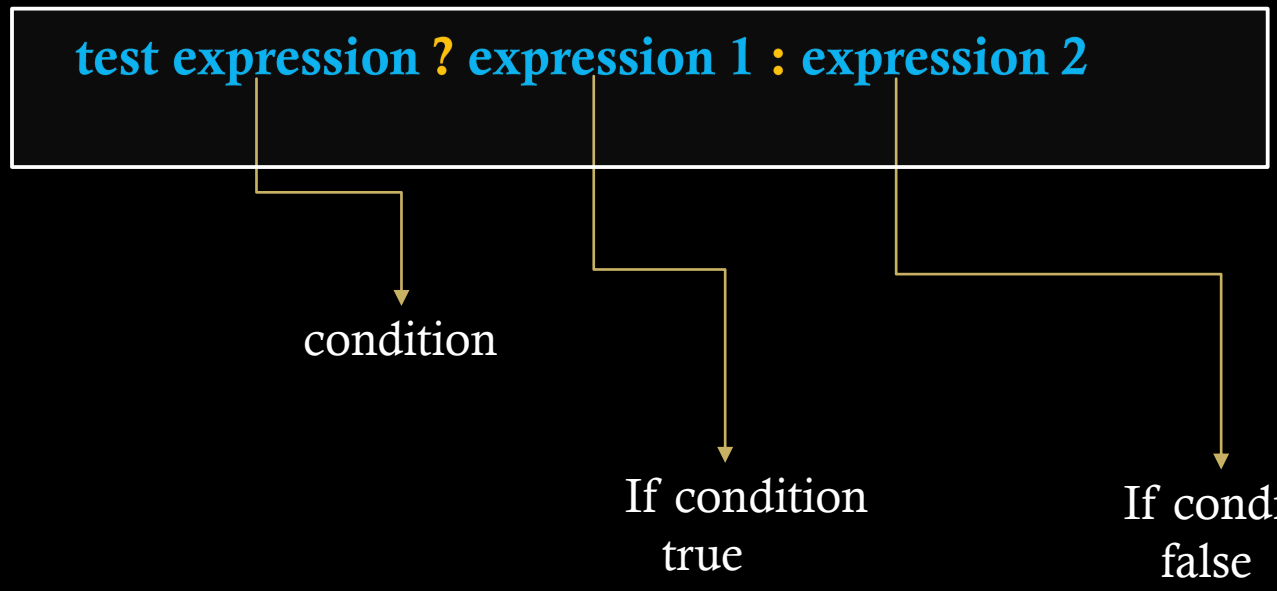
- (x>y) && (x>z) here this expression returns true if both condition are true
- (x>y) || (x>z) here this expression returns true if any one or both condition are true
- !(x>y) not operator reverses the state means if the condition is true it return false and if condition is false it returns true.

Example on logical operator

Go to code editor software

Conditional operator :-

- Conditional operator is ternary operator denoted by (**?** And **:**) which required three expression or operand and this is written as



Example on conditional operator

Q. Check 5>2 if it is true, so print true and if it is false print false using conditional operator

Go to code editor software

Bitwise operator :-

- The bitwise operator are used for operation on individual bits it operate on integer only

Operator	meaning
&	bitwise AND
|	bitwise OR
^	bitwise XOR
~	bitwise complement
<<	left shift
>>	right shift

- **Bitwise AND :-** Output of bitwise AND is 1 if the corresponding bits of two operands is 1 , if either bit of an operand 0 the result of corresponding bit evaluated to 0.

Eg . Bitwise AND of 12 and 25 are

12 = 00001100
25 = 00011001

```
  00001100  = 12
& 00011001 =  25
  --------------
  00001000  = 8
  --------------
```

- **Bitwise OR :-** Output of bitwise OR is 1 if at least one corresponding bits of two operands is 1 , otherwise result is 0

  Eg . Bitwise OR of 12 and 25 are

  12 = 00001100
  25 = 00011001

  ```
    00001100  = 12
  | 00011001 =  25
  ----------------
    00011101  = 29
  ----------------
  ```

- **Bitwise XOR :-** Output of bitwise XOR is 1 if the corresponding bits of two operands are opposite , otherwise result is 0

  Eg . Bitwise XOR of 12 and 25 are

  12 = 00001100
  25 = 00011001

  ```
    00001100  = 12
  ^ 00011001 =  25
  ----------------
    00010101 = 21
  ----------------
  ```

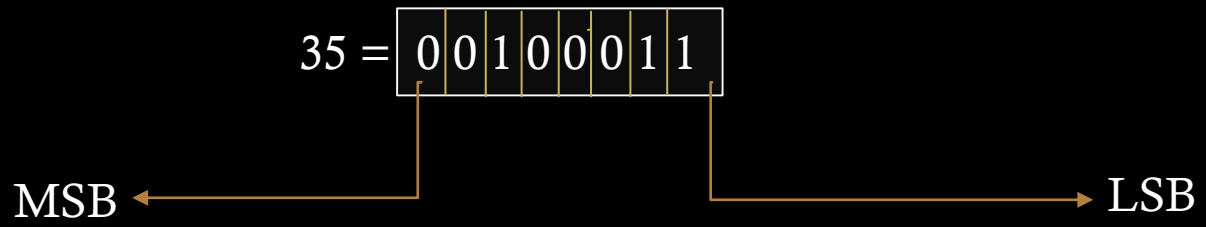
- **Bitwise compliment :-** It change 0 to 1 and 1 to 0

  Eg . Bitwise compliment of 35

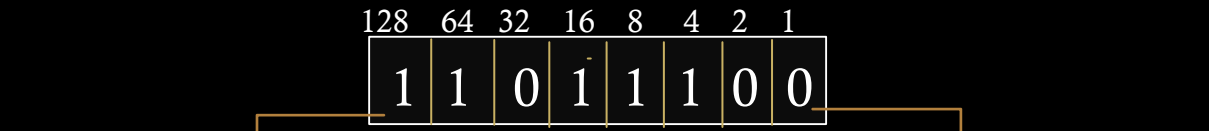
Most significant bit (MSB) → 0 for positive
                           → 1 for negative

Least significant bit (LSB) → 0 for even
                            → 1 for odd

Eg . Bitwise compliment of 35



one's compliment



- a

- 128 + 64 + 0+16+8+4+0
-64+16+8+4
-64+28
-36

Left shift

- Denoted as <<
- Eg.

n << i (n : first operand , i : second operand)

Left shift = $n * 2^i$

right shift

- Denoted as >>
- Eg.

n >> i (n : first operand , i : second operand)

Right shift = $n / 2^i$

Example of bitwise operator

Go to code editor software

Special operator

Symbol	Description	Example
&	It is used find address of a variable	int a=10; printf("%d",&a);
*	It is used to declare pointer type variable.	int *p;
sizeof()	It returns the memory size of variable.int	a=10; printf("%d",sizeof(a));

Operator precedence and associativity

C Operator Precedence

The following table lists the precedence and associativity of C operators. Operators are listed top to bot
descending precedence.

Precedence	Operator	Description	Associativity
1	++ --	Suffix/postfix increment and decrement	Left-to-right
	()	Function call	
	[]	Array subscripting	
	.	Structure and union member access	
	->	Structure and union member access through pointer	
	(*type*){*list*}	Compound literal(C99)	
2	++ --	Prefix increment and decrement[note 1]	Right-to-left
	+ -	Unary plus and minus	
	! ~	Logical NOT and bitwise NOT	
	(*type*)	Cast	
	*	Indirection (dereference)	
	&	Address-of	
	sizeof	Size-of[note 2]	
	_Alignof	Alignment requirement(C11)	
3	* / %	Multiplication, division, and remainder	Left-to-right
4	+ -	Addition and subtraction	
5	<< >>	Bitwise left shift and right shift	
6	< <=	For relational operators < and ≤ respectively	
	> >=	For relational operators > and ≥ respectively	
7	== !=	For relational = and ≠ respectively	

7	== !=	For relational = and ≠ respectively	
8	&	Bitwise AND	
9	^	Bitwise XOR (exclusive or)	
10	|	Bitwise OR (inclusive or)	
11	&&	Logical AND	
12	||	Logical OR	
13	?:	Ternary conditional[note 3]	Right-to-left
14[note 4]	=	Simple assignment	
	+= -=	Assignment by sum and difference	
	*= /= %=	Assignment by product, quotient, and remainder	
	<<= >>=	Assignment by bitwise left shift and right shift	
	&= ^= |=	Assignment by bitwise AND, XOR, and OR	
15	,	Comma	Left-to-right

example

Go to code editor software

Thank You !!

Dhanybad !!

Shukriya !!