Operator

This is a symbol use to perform some operation on variables, operands or with the constant. Some operator required 2 operand to perform operation or Some required single operation.

Several operators are there those are, arithmetic operator, assignment, increment , decrement, logical, conditional, comma, size of , bitwise and others.

1. Arithmatic Operator

This operator used for numeric calculation. These are of either Unary arithmetic operator, Binary arithmetic operator. Where Unary arithmetic operator required

only one operand such as +,-, ++, --,!, tiled. And these operators are addition, subtraction, multiplication, division. Binary arithmetic operator on other hand required two operand and its operators are +(addition), -(subtraction), *(multiplication), /(division), %(modulus). But modulus cannot applied with floating point operand as well as there are no exponent operator in c.

Unary (+) and Unary (-) is different from addition and subtraction.

When both the operand are integer then it is called integer arithmetic and the result is always integer. When both the operand are floating point then it is called floating arithmetic and when operand is of integer and floating point then it is called mix type or mixed mode arithmetic . And the result is in float type.

2.Assignment Operator

A value can be stored in a variable with the use of assignment operator. The assignment operator(=) is used in assignment statement and assignment expression. Operand on the left hand side should be variable and the operand on the right hand side should be variable or constant or any expression. When variable on the left hand side is occur on the right hand side then we can avoid by writing the compound statement. For example,

```
int x= y;

int  Sum=x+y+z;
```

3.Increment and Decrement

The Unary operator ++, --, is used as increment and decrement which acts upon single operand. Increment operator increases the value of variable by one .Similarly decrement operator decrease the value of the variable by one. And these operator can only used with the variable, but cann't use with expression and constant as ++6 or ++(x+y+z).

It again categories into prefix post fix . In the prefix the value of the variable is incremented 1st, then the new value is used, where as in postfix the operator is written after the operand(such as m++,m--).

EXAMPLE

let y=12;

z= ++y;

y= y+1;

z= y;

Similarly in the postfix increment and decrement operator is used in the operation . And then increment and decrement is perform.

EXAMPLE

let x= 5;

y= x++;

y=x;

x= x+1;

4.Relational Operator

It is use to compared value of two expressions depending on their relation. Expression that contain relational operator is called relational expression.

Here the value is assign according to true or false value.

a.(a>=b) || (b>20)

b.(b>a) && (e>b)

c. 0(b!=7)

5. Conditional Operator

It sometimes called as ternary operator. Since it required three expressions as operand and it is represented as (? , :).

SYNTAX

exp1 ? exp2 :exp3

Here exp1 is first evaluated. It is true then value return will be exp2 . If false then exp3.

EXAMPLE

```
void main()
{
 int a=10, b=2
  int s= (a>b) ? a:b;
  printf("value is:%d");
 }
```

Output:

Value is:10

6. Comma Operator

Comma operator is use to permit different expression to be appear in a situation where only one expression would be used. All the expression are separator by comma and are evaluated from left to right.

EXAMPLE

```
int  i, j, k, l;
for(i=1,j=2;i<=5;j<=10;i++;j++)
```

7. Sizeof Operator

Size of operator is a Unary operator, which gives size of operand in terms of byte that occupied in the memory. An operand may be variable, constant or data type qualifier.

Generally it is used make portable program(program that can be run on different machine) . It determines the length of entities, arrays and structures when their size are not known to the programmer. It is also use to allocate size of memory dynamically during execution of the program.

EXAMPLE

```
main( )

{

int sum;

float f;

printf( "%d%d" ,size of(f), size of (sum) );

printf("%d%d", size of(235 L), size of(A));

}
```

8. Bitwise Operator

Bitwise operator permit programmer to access and manipulate of data at bit level. Various bitwise operator enlisted are

one's complement	(~)
bitwise AND	(&)
bitwise OR	(|)
bitwise XOR	(^)
left shift	(<<)
right shift	(>>)

These operator can operate on integer and character value but not on float and double. In bitwise operator the function showbits( ) function is used to display the binary representation of any integer or character value.

In one's complement all 0 changes to 1 and all 1 changes to 0. In the bitwise OR its value would obtaining by 0 to 2 bits.

As the bitwise OR operator is used to set on a particular bit in a number. Bitwise AND the logical AND.

It operate on 2operands and operands are compared on bit by bit basic. And hence both the operands are of same type.

Logical or Boolean Operator

Operator used with one or more operand and return either value zero (for false) or one (for true). The operand may be constant, variables or expressions. And the expression that combines two or more expressions is termed as logical expression. C has three logical operators :

Operator	Meaning
&&	AND
||	OR
!	NOT

Where logical NOT is a unary operator and other two are binary operator. Logical AND gives result true if both the conditions are true, otherwise result is false. And logial OR gives result false if both the condition false, otherwise result is true.

Precedence and associativity of operators

Operators	Description	Precedence level	Associativity
()	function call	1	left to right
[]	array subscript		
→	arrow operator		
.	dot operator		
+	unary plus	2	right to left
-	unary minus		
++	increment		
--	decrement		
!	logical not		
~	1's complement		
*	indirection		
&	address		
(data type)	type cast		
sizeof	size in byte		
*	multiplication	3	left to right
/	division		
%	modulus		
+	addition	4	left to right

-	subtraction		
<<	left shift	5	left to right
>>	right shift		
<=	less than equal to	6	left to right
>=	greater than equal to		
<	less than		
>	greater than		
==	equal to	7	left to right
!=	not equal to		
&	bitwise AND	8	left to right
^	bitwise XOR	9	left to right
|	bitwise OR	10	left to right
&&	logical AND	11	
||	logical OR	12	
?:	conditional operator	13	
=, *=, /=, %= &=, ^=, <<= >>=	assignment operator	14	right to left
,	comma operator	15	