7/9/19 UNIT-2 OPERATORS

Operators:- An operator is a symbols that tells the computer to perform various mathematical calculations.

An operator is a symbols that operates on the data or variable.

'C' language has well defined operators to perform various operations on data.

'C' provides 8 types of operators.

1) Arithematic operators.
2) Relational operators
3) Logical operators.
4) Assignment operators.
5) Increment and decrement.
6) Conditional operator.
7) Bitwise operators
8) Special operators.

⇒ Arithematic operators:- 'C' provides 5 types of arithmetic operators. It is used to perform mathematical operations such as addition, substraction, multiplicat

Division and modulus division.

a=10; b=2

Operator	Meaning	Example
+	addition	a+b = 10+2=12
-	Substraction	a-b = 10-2=8
*	multiplication	a*b = 10*2=20
/	Division	a/b = 10*2=5
%	Modulus division	a%b = 0

a=10; b=3

a÷b = 10/3 = 3

a%b = 10%3 = 1

⇒ 9%2 → 1

9.0%2 → error

9%2.0 → error

9.0%2.0 → error

9/2 → 4 (Int/Int = Int)

9.0/2 → 4.5

9/2.0 → 4.5

9.0/2.0 → 4.5

(float/float = float)

⇒ +, -, *, / are integer float operator but % only for integer. It can't be used on floating point data it is a integer operator.

- 2/7 → 0 ; 2.0/7 → 0.22.
- -14/-7 = -2

  -14/7 = -2

  14/7 = 2

  14/-7 = 2

{ Sign of 1st operator is sign of the result. }

Example:-

1, Write a 'c' program to add, sub, mul & division.

```
#include <stdio.h>
void main( )
{
int a=10, b=2;
Printf ("a+b = %d\n", a+b);
Printf ("a-b = %d\n", a-b);
Printf ("a*b = %d\n", a*b);
printf ("a/b = %d\n", a/b);
Printf ("a%b = %d\n", a%b);
}
```

O/p :- a+b = 12
a-b = 8
a*b = 20
a/b = 5
a%b = 0

2, Write a 'c' program to convert no. of days into months and dates.

Ans
```
#include<stdio.h>
void main()
{
int months, days;
printf("enter no. of days");
scanf("%d", &days);
months = days/30;
days = days%30;
printf("months = %d days=%d", months,days);
}
```

Q19:- 9|9|19 = *Relational operator :- * ==

A rational operator checks relationship between two operants (variables ; constant). If the relation (condition) is true it written 1 (true). If the condition is false it written '0'.

A relationaloperators are used for comparison of values of two variables (to compaid two quantities values). It is used in decision making statements or in loops.

'C' provides 6 types of relational operators

>, <, >=, <=, ==, !=

Operator	Description	example	retain value
>	Greater than	10>5	1 (true)
<	less than	10<5	0 (false)
>=	greater than or equal to	10>=5	0 (false)
<=	less than or equal to	10<=5	0 (false)
==	equal to	10==5	0 (false)
!=	not equal to	10!=5	1 (true)

Q, Write a 'c' programme to greater, less, greater than or equal to, less than or equal to, equal to and not equal to.

```
#include <stdio.h>
Void main()
{
int a=5, b=2;
Printf("a>b=%d\n", a>b);
printf("a<b=%d\n", a<b);
printf("a>=b=%d\n", a>=b);
Printf("a<=b=%d\n", a<=b);
Printf("a==b=%d\n", a==b);
Printf("a!=b=%d\n", a!=b);
}
```

O/p:- a>b = 1
a<b = 0
a>=b = 0
a<=b = 0
a==b = 0
a!=b = 1

*Logical operators:- *

It is used to check/test more than one condition and make decision.

(or)

It is used to evaluate two or more conditions.

⇒ It is used to combine two or three relational expressions. 'C' provides 3 types of logical operators.

1) Logical And (&&)
2) Logical OR (||)
3) Logical NOT (!)

1) Logical And (&&) :- It performs logical multiplication

2) Logical OR (||) :- It performs logical addition.

And truth table

OP_1	OP_2	Result
T	T	T
T	F	F
F	T	F
F	F	F

* OR truth table

OP_1	OP_2	Result
T	T	T
T	F	T
F	T	T
F	F	F

* Not truth table:-

OP_1	value
T	F
F	T

-

Operator	Meaning	example	result
&&	logical AND	7>=3 && 1<5 (T T)	1
||	logical OR	7>10 || 1<3 (F T)	1
!	logical Not	!(7==7)	0

```
#include <stdio.h>
void main()
{
int a=6 ; b=2;
Printf("a>b= %d\n",a>b);
```

11/9/19 (4) Assignment operators

- It is used to assign a value to a variable. (or) It is used to assign the result of an expression to the variable.

*Syntax of the Assignment operators are;-

Variable name operator constant / variable / expression ; (or) v = exp ;

v → variable, exp → expression

Eg:- a=5 ; b=a ; c=a+b; c=10; b=5;

	Operative	meaning	Example	short hand Operative
1,	=	• Assign	a=b	a=2; b=5 → a=2
2,	+=	• increament then assign	a+=b	a=a+b → a=5+2=7
3,	-=	• Decreament then assign	a-=b	a=a-b = 5-2=3
4,	*=	• multiply then assign	a*=b	a=a*5 = 5*2 = 10
5,	/=	• Division then assign	a/=b	a=a/b = 2/5
6,	%=	• modulus division then assign	a%=b	a=a%b = 5%2=1

2, Write a 'c' program to assignment operators

```
#include<stdio.h>
void main()
{
int a=10, b=2;
a+=2;
Printf("a+=%d\n", a);
a-=2;
Printf("a-=%d\n", a);
a*=2;
Printf("a*=%d\n", a);
a/=2;
Printf("a/=%d\n", a);
a%=2;
printf("a%=%d\n", a);
}
```

Op :-

a = a+2 = 12

a = a-2 = 8

a = a*2 = 20

a = a/2 = 5

a = a%2 = 0

⑤ Increment / Decrement Operators :-

- Increment & Decrement operators are used to change the value of an operator constant & variable by '1'. Those operators are called unary operators.

=> C provide 2 types of increment & decrement operators are ++ & --.

x = 20 ; y = 10

- z = x * y++ = 200 + 20
  = 220
- a = x * y = 20 * 11 = 220
- z = x * ++y = 200 + 20
  = 220
- a = x * y = 220

i.e.,

Operator	meaning
++	Increment
--	Decrement

=> Increment (++) operator :- It increases the value by 1. (or) It adds 1 to the operants.

=> Decrement (--) operator :- It decreases the value by 1. (or) It subtracts 1 value from the operants.

Eg:- { ++a (or) a++ => a=a+1 (or) a+=1
       --a (or) a-- => a=a-1 (or) a-=1 }

(1)

=> a=10	a=10
a++	++a
a=?	a=?
11	11

(2)

=> a=5	a=5
b=++a	b=a++
b=6	b=?
a=6	a=6

*Preincrement (or) Prefix:-

When prefix {++} (or) {--} is used an expression, ~~when~~ the variable is incremented or decremented by 1 and then the expression is evaluated using the new value of variable.

• Post increment (or) suffix.

When postfix {++} or {--} is used an expression, the ~~variable~~ expression is evaluated 1st using the original value of the variable and then the variable is incremented or decrement by

Eg:- Program

```
#include <stdio.h>
void main()
{
int a=10, b=5, c=, d;
c= a++ - b;
Printf ("a = %d\n b= %d\n c= %d\n", a,b,c);
d= ++a +b;
printf ("a= %d\n b= %d\n d= %d", a,b,d);
}
```

O/p:-

① a = 10, b = 5, c = 5

② a = 12, b = 5, d = 17

⇒ Conditional operator:-

It ~~used~~ called as ternary operator. (It takes 3 operants). It is used to construct conditional expression or evaluate expression.

- C provides only 1 operator:- operator means conditional Operator.

? :

• The conditional operator takes form syntax

* Syntax : Variable = (conditional)? exp1 : exp

(or)

Variable = exp1 ? exp2 : exp3;

=* It contains conditions and then follows by 2 values / expressions.

• The conditional operator written by 1 values, if the condition is true and returns ~~written~~ by another value, if the condition is false.

• It returns only 1 value at a time.

ex: c = 2>3 ? 10 : 5;

c = 5

c = 2<3 ? 10 : 5;

c = 10

• Write a c programe to find greatest of two numbers using conditional operator

```
#include <stdio.h>
Void main ()
{
int a, b, c;
Printf ("enter the a b values\n");
```

```
scanf ("%d%d", &a, &b);
c = (a>b)?a:b;
printf ("the greater no. of the f is %d", c);
}
```

Output a = 5 ; b = 2.

∴ c = 5 > 2 ? 5 : 2

c = 5

- Write a c programe to find greatest of 3 numbers using conditional operators.

```
#include <stdio.h>
void main ( )
{
int 4, 2, 5, a, b, c, d, e;
Printf ("enter the a b values);
scanf ("%d %d", &a, &b);
d = (a>b)?a:b;
Printf ("enter the d, c values);
Scanf ("%d %d", &d, &c);
e = (d>c)?d:c
Printf ("the greater no. of thef is %d", e);
}
```

Bitwise Operator:-

It is used to perform bitlevel operations (low level operation). It is used for manupulation of data at bit level. These operaters works on bits (0 & 1) and perform to bitlevel operations.

- C provides 6 types of Bitwise operators

Operator	Mean
&	Bit and
|	Bit or
^	~~Bit~~ XOR (exclusive or)
~	Bitnot (1's complement)
>>	Rightshift
<<	leftshift

*Truth table:-

A	B	A&B	A|B	A^B	~A
0	0	0	0	0	1
0	1	0	1	1	1
1	0	0	1	1	0
1	1	1	1	0	0

Ex:- a=8 ; b=4

* a&b ⟶ 8 & 4 =

Example :- a = 8, b = 4.

Q. Write a 'c' program on bitwise And

```
#include <stdio.h>
Void main ()
{
int a,b,c;
Printf ("enter the values\n");
Scanf ("%d %d", &a, &b);
c= a&b;
Print ("the result is %d \n", c);
}
```

O/p:- 4 & 3 → 0

4 | 3 → 7

8 | 4 → 12

2 | 8 – 0
2 | 4 – 0
2 | 2 – 0
1 – 0

2 | 4
2 | 2 – 0
1 – 0

1000
0100
1100.
$2^3 \; 2^2 \; 2^1 \; 2^0$

$\Rightarrow 2^3 \times 1 + 2^2 \times 1 + 2^1 \times 0 + 2^0 \times 0$

$= 12$

Program on bitwise OR

```
#include <stdio.h>
void main()
{
int a,b,c;
Printf ("enter the values\n");
scanf ("%d%d", &a,&b);
c = a|b;
Printf ("the result is %d\n", c);
}
```

O/P:- 8 | 4 → 12

Program on bitwise XOR

```
#include<stdio.h>
void main()
{
int a, b, c;
Printf ("enter the values\n");
scanf ("%d%d", &a, &b);
c = a^b;
Printf ("the result is %d\n", c);
}
```

O/P:-

8 ^ 4 → 12

Program on bitwise Not

```
#include <stdio.h>
void main( )
{
int a, b;
printf ("enter the values \n");
scanf ("%d", &a, );
b= ~a;
Printf (" the result is %d \n", b);
}
```

O/P :- ~8 → 7

8 → 1000

~8 0111

2^3 2^2 2^1 2^0

$\Rightarrow 2^3 \times 0 + 2^2 \times 1 + 2^1 \times 1 + 2^0 \times 1$

$= 4 + 2 + 1$

$= 7.$

Program on bitwise right shift (>>)

```
#include <stdio.h>
void main()
{
int a,b;
Printf ("enter the 'a' value \n");
Scanf ("%d", &a);
b = a>>2;
Printf ("the right shifted data is %d", b);
}
```

formula:

$$b = \frac{n}{2^s}$$

n → given, s → shifted

O/p:-

a>>2 | a=8

2 | 8 — 0
2 | 4 — 0
2 | 2 — 0
1 — 0

0 0 0 0 1 0 0 0

0 0 0 0 0 0 1 0

$2^7\ 2^6\ 2^5\ 2^4\ 2^3\ 2^2\ 2^1\ 2^0$

$\Rightarrow 2^7 \times 0 + \dots + 2^2 \times 0 + 2^1 \times 1 + 2^0 \times 0$

$= 2$

Program on bitwise left shift (<<)

```
#include <stdio.h>
void main()
{
int a, b;
Printf ("enter the value \n");
scanf ("%d", &a);
b = a<<2;
Printf ("the left shifted data is
        %d", b);
}
```

formula:

$b = n * 2^s$

n = no. of bits given

s = shifted

O/P:-

a<<2 — a=8

00001000

00001000000

$2^5\ 2^4\ 2^3\ 2^2\ 2^1\ 2^0$

$\therefore 2^5 \times 1 = 32$

7 & 3 → 3

7 | 3 → 10

7 ^ 3 → 10

~7 → 1

7 >> 3 → 7/8

7 << 3 → 56

18/9/19

* Special Operators:-

'C' supports some special operators they are.

1. comma operator
2. sizeof operator

* Comma operator(,):-

It is used to seperate two or more expression / values.

Eg:- int a, b, c;

int a=10, b=20, c=30;

Printf ("%d %d %d", 5+3, 5-3, 5*3);

*sizeof operator :-

It ~~is written as~~ returns / give the size of data type of variable.

Eg :- ① int a = 10;
printf("%d", sizeof(a)); (2).
printf("%d", a); (10).

⇒ Program on sizeof operator :-

```
#include <stdio.h>
void main()
{
printf("%d", size_of (char) ); = 1
printf("%d", size_of (short)); = 1
printf("%d", sizeof (int)); = 1
printf("%d", sizeof (long) ); = 2
printf("%d", sizeof (float)); = 4
printf("%d", sizeof (double)); = 8
printf("%d", sizeof (long double)); = 10
}
```

O/p :- 1
1
1
2
4
8
10

Operator	Symbol representation	
Arithematic operator	+, -, *, \, -/.	
• Relational operator	==, <, >=, <=, !=	Binary
• Logical operator	&&, ||, !	Binary
• Assignment	=, +=, -=, *=, /=, %=	Binary
• Increment & Decrement	++, --	→ Unirary
• Conditional operator	?:	
• Bitwise operator	&, |, ^, ~, >>, <<	ternary
• Special operator	, sizeof	ternary

```
#include<stdio.h>
void main()
{
int a, b, c, d;
a = 15; b = 10;
c = ++a - b;          → 10        c = 16 - 10 = 6
printf("a = %d \n b = %d \n c = %d \n", a, b, c);
d = b++ + a;          10 + 16 = 26
printf("a = %d \n b = %d \n d = %d \n", a, b, d);
                                          16  11  26
printf("%d", a/b);    1
printf("%d", a%b);    5
printf("%d", a>=b);   0
printf("%d", a>b && c>d);   0
printf("%d", a*=b);   16 * 11 = 126
printf("%d", a&b);    16 & 11
printf("%d", a>>3);   2
printf("%d", c>d ? 1 : 0);   6 > 26 = 0
printf("%d", sizeof(a));   2
}
```

① write a c program to swapp two numbers using 3rd variable?

```
#include<stdio.h>
void main()
{
int a,b,t;
Printf ("enter the values\n");
scanf( "%d %d", &a, &b);
Printf ("befor swapping a=%d\n b=%d\n",a,b
t= a;
a =b;
b=t;
Printf ("after swapping a=%d\n b=%d\n",a,b
}
```

Op:- a=10 b=20

t=a
t=10
a=20
b=10

a	b	t
20	10	10

② write a c program to swap two numbers without using 3rd variable?

```
#included <stdio.h>
void main()
{
int a,b,t;
```

Printf ("enter the values\n");
scanf ("%d %d", &a, &b);
printf ("before swapping a=%d\n b=%d\n", a, b);
a = a+b;
b = a-b;
a = a-b;

o/p:- b = a-b
b = 30-20
b = 10

a = a-b
= 30-10
a = 20

=> a = a+b
a = 10+20
a = 30

∴

a	b
20	10

③ Write a 'C' programme to find whether the number is positive or negative.

```
#include <stdio.h>
void main()
{
int a,b;
Printf (" enter the values n \n");
Scan f ("%.d ", &n);
Printf (" (n>0)? printf (" even no. "): Printf("odd");
}
```

8Q, write a c programe to find whether a person is eligible or not.

```
#include <stdio.h>
void main()
int n,x;
printf ("Enter n and x in");
scanf ("%d %d", &n, &x);
(n>=18)? Printf ("
Printf (n>=18) ? (eligible): Prinf (a
                                    eligib
}
```

O/P enter -16
not eligible.

25/9/19 **Expression Evaluation in C**

In C programming the expression evaluation is based on operator ① president and ② operator Associativity.

An expression is a combination of operators and operants (constants & variable) that reduces to a single value.

Syntax:- Variable = expression

Example:- $x = a+b*c$ ⇒ $y = a*b+c$ $\left\{\because \begin{matrix} a=3 \\ b=2 \\ c=1 \end{matrix}\right\}$

$z = a/b+c-d$

$c = 5*3+4/2-1$

$= 15+2-1$

$\boxed{c = 16}$

Operator President & Associativity:-

① President:- It defines the priority of evaluation of operators in an expressions.

② Associativity:- It defines the odour in which the same president operators are evaluated in an expressions.

hierachy of Arithematic operator.

* / o/. → 1st priority

+ & - → 2nd priority.

⇒ *, /, o/. are has highest priority than + & -.

⇒ Binary operators are evaluated from left to right.

⇒ Unirary operators are evaluated from right to left.

Operators with president & Associativity:-

Operator	Associativity	Priority.
1. ( ) (functions & paratices)	left to right.	1st
[ ] (Arrays (or) brackets)		1st
2. ++	right to left.	2nd
-- & Size of		
3. * / o/.	left to right	3rd
4. + -		4th.
5. >, <, >=, <=	left to right	5th.
==		

6. &&, ||, ! left → right 6th

7. &, |, ^ left → right 7th

8. ?, : left → right 8th

9. Assignment right → left 9th

(=, +=, -=, *=, %=, /=)

Example

- a/b + c * d = 3/2 + 1 * 4 { ∵ a=3, b=2, c=1, d=4 }

= 1 + 1 * 4

= 1 + 4

= 5

- a - a/5 * 5 % 3 > a % 5 % 3 =

~~= a - 1 * 0 > 0~~ 7 > 1

25 | 9 | 19

x = a - b / 3 + c * 2 - 1 = 10

y = a - b / (3 + c) * (2 - 1) = 7

z = a - (b / (3 + c) * 2) - 1 = 4 { a=19, b=12, c=3 }

→ 17 - 8 / 4 * 2 + 3 - ++a

= ~~23~~ - ++a 17 - 2 * 2 + 2 + 3 - 6

~~12~~ 13 + 3 - 6

10 11

a=5

* 8 / (2*(2*2)) = 8/8 = 1

* 5 x 4 + 8/2 = 20 + 4 = 24.

* b/c * a = 36 @ 12/3 * 9 = 36

= 36 - 36

= 0 //

Q) To evaluate c expression

```
#include<stdio.h>
void main()
{
int a,b,c;   a=9, b=12, c=3;
float x,y,z;
Printf (
```

```
#include<stdio.h>
void main()
{
int a=9, b=12, c=3;
float x, y, z;
x= a-b/3 +c*2-1;
y= a-b/(3+c)*(2-1);
z= a-(b/(3+c)*2)-1;
Printf ("x= %f\n", x);
Printf ("y= %f\n", y);
Printf ("z= %f\n", z);
Scanf ("%d %d %d %f %f %f", &a, &b
```

&c, &x, &y, &z);

}.

*Type Conversion (or) Type casting:-

Type conversion is also called type casting. Type conversion refers to changing an variable of one data type into another data type is called type conversion.

There are two of type conversion.

① Implicit type conversion:-

It is called as automatic type casting / conversion. It performes / done by the compiler. The compiler will automatically change or convert one type of data to another type.

Eg:- int a = 20.5;
printf ("%d", a);
~~float f~~ "20"

Eg:- float f=20
printf("%f",f);
"20.0"

Comments. int = char

char c = 65;
Printf ("%c", c); A.

Explicit type conversion.

to print ASCI value.

```
#include <stdio.h>
void main()
{
char c = 'x';
printf ("%d", c);
}
```

26/9/19

Explicit type of conversion:-

It is perform by the programmer (user).

Syntax:- (datatype) expression;

Example:-

* int a = 7.0 % 2; | o/p (error)
  printf("%d", a);

* int a = (int) 7.0 % 2.;
  printf ("%d", a); | o/p E.TC from float to int.

* int a = 7, b = 2;
  float c = (float) a/b; | o/p E.TC from int to float = 3.5
  Printf ("%f", c);

* float a=45.5, b=5.5, c=6.5;
int d = (int)a + (int)b + (int)c;

Type casting meaning :-

Type cast	Meaning
(int) 7.0;	— 7.0 converted to int.
(int) a+b;	— 'a' value is converted to int then added to 'b'.
(int) (a+b);	— the result of expression is converted to int.

TEST

1Q, a, Define a flow chart with example?

(or)

b, Define an algorithm with example?

2Q a, Define pseudo code? write pseudo code addi of 2 num.

(or)

b) Define token & list out types of tokens

3Q, a, Define Identifier & write rules.

(or)

b, Defference b/w Compiler & Interpreter

4Q, a, Evalute $9 - 9/5 * 5 \% 3 > 9 \% 5 \% 3$

(or)

b, Give the formatt of conditional operato & when it is used.

5Q, a, Type conversion with example?

(or)

b, Imp of presidence & associativity?

6Q, a, Explain Increament & Decrement with ex?

(or)

b, Size of operator with example program

C language is used for develop desktop application & system software.

Some applications of 'C' language are

1. 'C' programing language can be use to design the system software like operating system (window & Index).
2. To develop system software like data base (sql, my sql).
3. It can be to design network devices.
4. Used in writting embedded software & also to develop compiler, assemblers, & interpreters.