OPERATOR	MEANING	ASSOCIATIVITY	l → LEFT r → RIGHT
( ), [ ], ·, ->	function, array, structure, structure	l, r	
-, +, ++, --, !, ~, &, *, sizeof ( ), (type)	Unary minus, unary plus, incr & dec, logical not, once complement, address, Value at address, size of, typecasting	r, l	
/, *, %	Division, Multiplication, Modulus	l, r	
+, -	Addition, subtraction	l, r	
<, >, <=, >=, ==, !=	less than, greater than, less equal, > equal, equals to, not equal	l, r	
&	Bitwise AND	l, r	
^	Bitwise XOR	l, r	
		Bitwise OR	l, r	
&&	logical AND	l, r	
||	logical OR	l, r	

OPERATOR	MEANING	ASSOCIATIVITY
? :	conditional	R, L
=, +=, -=, *=, /=, %=, &=, !=, ^=, <<=, >>=	Assignment	R, L
,	comma	L, R
;	semicolon	L, R

Ex:-

```
main ( )
{ int x ;
    x = 4 + 6/3 * 2 - 2 + 7 % 3;
    printf ("%d", & x);
}
```

O/P:- 7

Expt. Solution:-

$4 + 2 * 2 - 2 + 7 \% 3$

$= 4 + 4 - 2 + 7 \% 3$

$= 4 + 4 - 2 + 1$

$= 8 - 2 + 1$

$= 6 + 1$

$= 7$

EX:-
```
main ( )
{ int a, b, c;
    a = b = c = 5;
    printf (" %d %d %d", a, b, c);

}
```

EX:-
```
main ( )
{ int a = 5, b = a+2, c = a+b;
    printf (" %d %d %d", a, b, c);

}
```

CONTROL STATEMENTS =>

1. Sequential

2. Branch Control

3. Switch

4. Loop

5. Jump Control