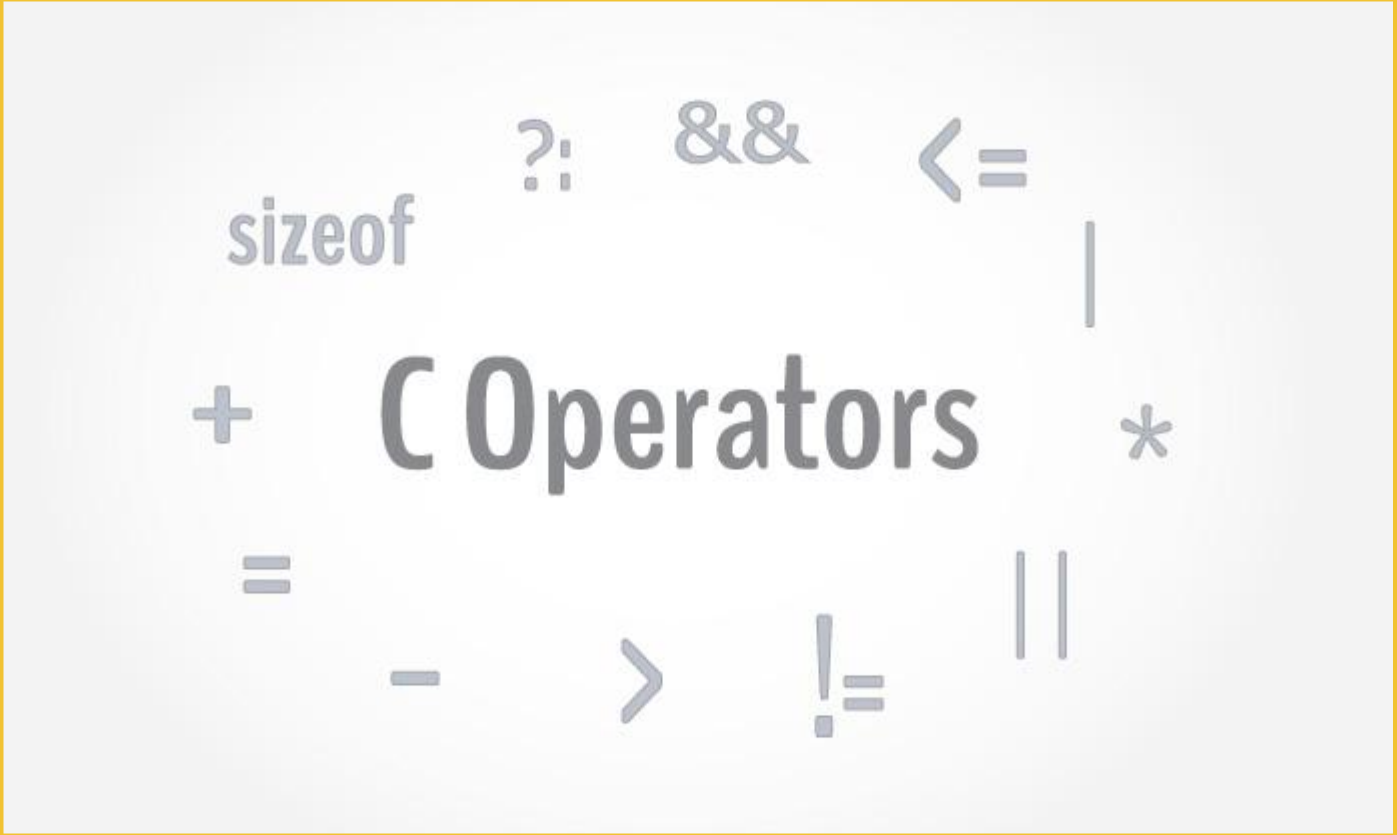


?:
&&
<=
sizeof
|
+
C Operators
*
=
||
-
>
!=

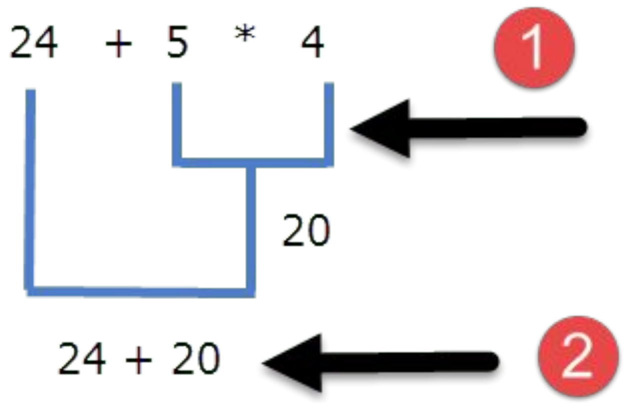
WHAT IS OPERATORS AND ITS ASSOCIATIVITY?

Operator	*Description*	*Associativity*
() [] . -> ++ --	Parentheses or function call Brackets or array subscript Dot or Member selection operator Arrow operator Postfix increment/decrement	left to right
++ -- + - ! ~ **(type)** * **&** **sizeof**	Prefix increment/decrement Unary plus and minus not operator and bitwise complement type cast Indirection or dereference operator Address of operator Determine size in bytes	right to left
* / %	Multiplication, division and modulus	left to right
+ -	Addition and subtraction	left to right
<< >>	Bitwise left shift and right shift	left to right
< <= > >=	relational less than/less than equal to relational greater than/greater than or equal to	left to right
== !=	Relational equal to and not equal to	left to right
&	Bitwise AND	left to right
^	Bitwise exclusive OR	left to right
|	Bitwise inclusive OR	left to right
&&	Logical AND	left to right
||	Logical OR	left to right
? :	Ternary operator	right to left
= += -= *= /= %= &= ^= |= <<= >>=	Assignment operator Addition/subtraction assignment Multiplication/division assignment Modulus and bitwise assignment Bitwise exclusive/inclusive OR assignment	right to left
,	Comma operator	left to right

Q.1) 24 + 5 * 4 ?

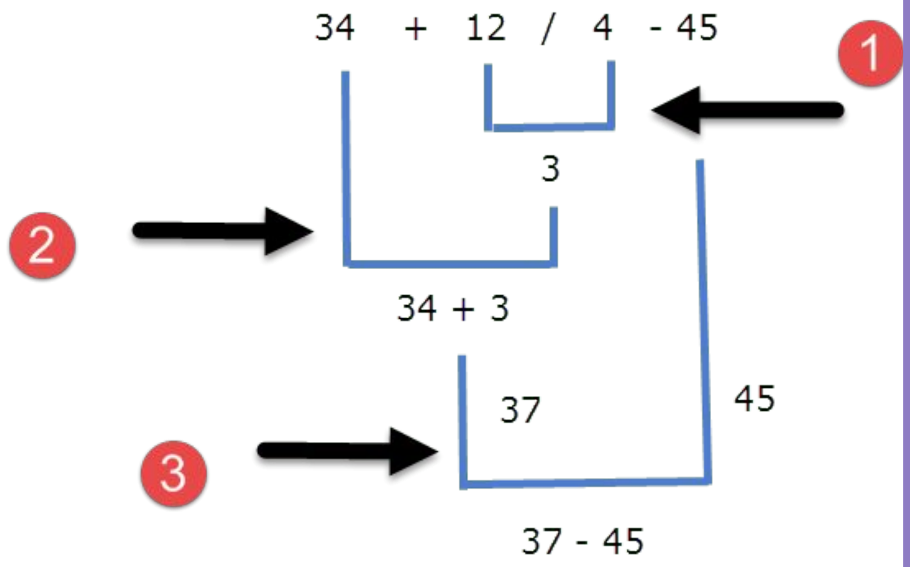
Q.2) 34 + 12/4 - 45 ?

Q.1)



Ans : 44

Q.2)



Ans : -8

P.1) $12 + 3 - 4 / 2 < 3 + 1$?

P.2) $13 + 2 - (4 / 2 < 3) + 1$?

P.1) 12 + 3 - 4 / 2 < 3 + 1 ?

O/P : 0

P.2) 13 + 2 - (4 / 2 < 3) + 1 ?

O/P : 15

```
int i = -5;

int l = i / 4;

int k = i % 4;

printf("%d %d\n", l, k);
```

```
int i = 5;

int l = i / -4;

int k = i % -4;

printf("%d %d\n", l, k);
```

```
int i = -5;

int l = i / -4;

int k = i % -4;

printf("%d %d\n", l, k);
```

```
int i = -5, k=2, l;

l=2+10*i/k-k%2;

printf("%d %d\n", l, k);
```

```c
int i = -5;

int l = i / 4;

int k = i % 4;

printf("%d %d\n", l, k); // -1 -1
```

```c
int i = 5;

int l = i / -4;

int k = i % -4;

printf("%d %d\n", l, k); //-1 1
```

```c
int i = -5;

int l = i / -4;

int k = i % -4;

printf("%d %d\n", l, k);// 1 -1
```

```c
int i = -5, k=2, l;

l=2+10*i/k-k%2;

printf("%d %d\n", l, k); // -23 2
```

INCREMENT /DECREMENT OPERATORS

PRE Increment/Decrement

1.) Increase/Decrease

2.) Assign value

```
int x=0, y;

y= x++ + ++x;

printf(“ %d %d \n”,x,y);

y=++y + ++x;

printf(“ %d %d \n”,x,y);
```

POST Increment/Decrement

1.) Assign value

2.) Increase/Decrease

```
int x=1, y;

y= x++ + x++ + --x;

printf(“ %d %d \n”,x,y);

y=y-- + y++;

printf(“ %d %d \n”,x++,y);
```

PRE Increment/Decrement

1.) Increase/Decrease

2.) Assign value

```
int x=0, y;

y= x++ + ++x;

printf(“ %d %d \n”,x,y); o/p : 2 2

y=++y + ++x;

printf(“ %d %d \n”,x,y); o/p : 3 6
```

POST Increment/Decrement

1.) Assign value

2.) Increase/Decrease

```
int x=1, y;

y= x++ + x++ + --x;

printf(“ %d %d \n”,x,y);  o/p : 2 5

y=y-- + y++;

printf(“ %d %d \n”,x++,y);  o/p : 2 9
```

Assignment operator

+= **-=**

***=** **/=**

%= **|=**

```
int a=1, b=2;

a += 1;

b -= a;

printf(“ %d %d \n”,a,b);

a *= a + b +1 ;

b %= 2+a*2;

printf(“ %d %d \n”,a,b);
```

Assignment operator

+= **-=**

***=** **/=**

%= **|=**

```
int a=1, b=2;

a += 1;  //2

b -= a;  //0

printf(“ %d %d \n”,a,b); o/p : 2 0

a *= a + b +1 ;  //6

b %= 2+a*2;  //0

printf(“ %d %d \n”,a,b); o/p: 6 0
```

Relational Operator

==
!=
<
>
<=
>=

1) 2 == 4%2
2) 4 < 4
3) 3/2 < 1
4) 1 == 2 != 3
5) 1 == 2 != 0

Relational Operator

== **!=**

< **>**

<= **>=**

```
1)    2 == 4%2      // 0

2)    4 < 4         // 0

3)    3/2 < 1       // 0

4)    1 == 2 != 3   // 1

5)    1 == 2 != 0   // 0
```

```
int  a=4 , b=7;

a = a + b - ( b = a);

printf(“ %d %d \n”,a, b);
```

```
int  a=4 , b=7;

a = a + b - ( b = a);

printf(“ %d %d \n”,a, b);


    O/P: 7  4
```