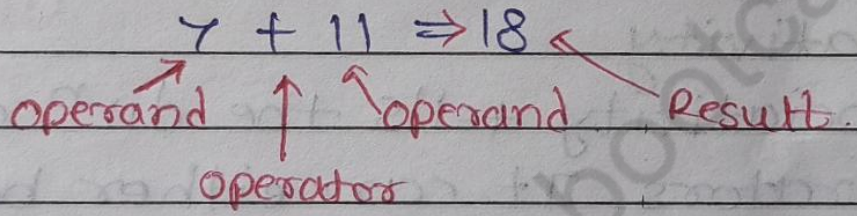
Operators and Expression.

Operators are used to perform operations on variables and values.



- Types of operator.

- Arithmetic operator ⇒ +, -, *, \, %, ++, --
- Assignment operator ⇒ =, +=
- Comparison operator ⇒ ==, <, <=, >, >=, !=
- Conditional operator (logical) ⇒ ||, &&
  - || → conditional ~~add~~ OR
  - && → conditional AND
- Logical operator(!) This converts value to boolean value.
- Bitwise operator ~~&&~~, ~~||~~, ~~^~~ &, |, ^
  - & → AND
  - | → OR
  - ^ → xOR

! Arithmetic operators can't work on booleans.
! % operator can work with float & double.
(Modula operator writtens remainder)..

☆ Precedance of operators.

☆☆ The operators are applied and evaluated based on precedence.

Ex. (+,-) have less precedence compair to (*,/) hence (*,/) will get evaluated first.

* In case we like to change this order we use paranthesis which have highest precedance.
e.g $\frac{x-y}{2} \Rightarrow (x-y)/2$

⊙ Associativity
Associativity tells the dirⁿ of execution of operators. It can either be left to right or right to left.
* / → L to R
+ - → L to R
=, +=, -= → R to L

⊙ Resulting data type after arithmetic oprⁿ

R = b+s ⇒ int	b ⇒ byte
R = s+i ⇒ int	s ⇒ short
R = l+f ⇒ float	l ⇒ long
R = i+f ⇒ float	f ⇒ float
R = c+i ⇒ int	i ⇒ int
	c ⇒ char
	d ⇒ double

R = c + s ⇒ int
R = i + d ⇒ double
R = f + d ⇒ double

⊙ Increment and decrement operators.
a++, ++a → Increment operators.
a--, --a → Decrement operators.

! If sign (++, --) comes first then operation gets done then value is assigned and if sign (++, --) comes after then value get assigned first then operation is done.

! During this datatype does not changes.
! This will work on all datatypes except booleans.

? working of bitwise operator.

& 11 → 0000 1011
22 → 0001 0110
0000 0010

	128	64	32	16	8	4	2	1
22	0	0	0	1	0	1	1	0

16+4+2 = 22

0 → false
1 → True

⇒ 2

& → Both true then op is True
| → ~~Both or~~ any one need to true then it will written True
^ → writtens difference

11 → 0000 1011
22 → 0001 0110
0001 1111 →

128	64	32	16	8	4	2	1
0	0	0	1	1	1	1	1

1+2+4+8+16 ⇒ 31

```
 11   0000 10110
^22   0001 0110
-------------------
      0001 1101
```
← writtens diff.

```
 11   0000 1011
^22   0001 0110
-------------------
      0001 1101
```
← It is an type of diff.

* Normal numbers have base 10.
* Bit wise operator works bit wise or works on binary numbers on "0" and "1"

Increment and decrement operators.

⊙ Increment

* Pre increment	* Post increment
change value first → ++a → Then assign value.	First assign value. → a++; → then change / increment value.

```
int a=10;
int b=0;
b = a++;
s.o.u.t(a); //11
s.o.u.t(b); //10
```

```
int a = 10;
int b = 0;
b = ++a;
s.o.u.t(a); //11
s.o.u.t(b); //11
```

⊙ Decrement.

- Predecrement	- Post decrement
First change value ← --a; → Then assign	First assign value ← a--; → Then change value.

```
int a=10;
int b=0;
b = --a;
sout(a); //9
sout(b); //9
```

```
int a =10;
int b =0;
b= a--;
sout(a); // 9
sout(b); // 10
```