# Unary Operators

```
let a = 10;
a++;
```

```
let a = 10;
let y = a++;
```

```
let a = 9;
let y = ++a;
```

++ → unary increment

-- → unary decrement

a--;

--a

a++ → increment by 1 unit ← ++a

increment first & then assign

a++ : postfix unary addition → increment after assignment

++a : prefix unary addition

let a = 10;

y = a++

→ i) → we first assign value of a to y. (i.e. old value)

ii) increment a by 1.

let a = 10

let y = ++a;

→

i) → we first increment the value of a by 1

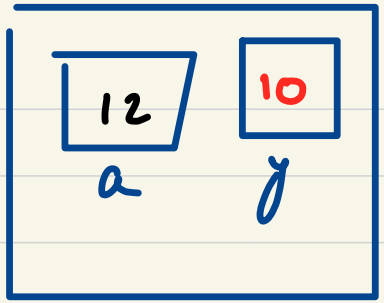
ii) → assign value of a to y.

```
let a=10 ;
let y=a++ ;
console.log(a,y);
```

11 10

```
a++;   // we are not assigning
console.log(a,y);
```

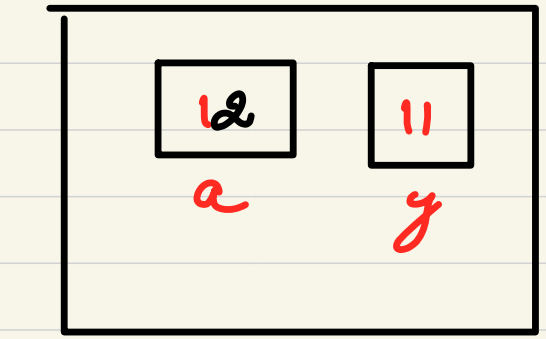
12 10



let a = 10 ;

let y = ++a;

Console.log (a, y) →




11 11

++a;

console.log (a, y)

12 11

+ (unary plus) → it tries to convert the variable into a number if it is not already a no.

+x

let x = "22";

let y = +x;

typeof y → Number

typeof x → String

it does it change the original operand but instead returns a converted value

unary minus → (-)

y = -x

→ it also converts the operand to ano, but always make the result after negate it.

let x = "22"

let y = -x;

console.log(x, y) → "22" -22

"22" → string

-22 → Number

x = "-22" → -22

y = -x; → 22

10

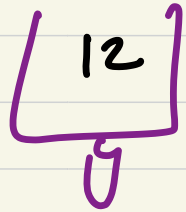
typeof → unary operator

typeof x → "Number"

let y = undefined;

typeof y → "undefined"

# Do-while Loops → even if the condition is false for start only it will still execute body/block of do once.

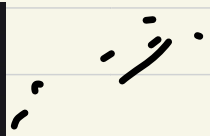



```
let y = 10;
do {
    y++;
} while(y < 12);
console.log(y)
```

keyword

```
let y = 10;
while ( y < 5) {
    y++;
}
console.log (y)
```

```
let y = 10; // y -> 15
do {
  y++;
  console.log("inside do", y);
} while(y < 15);
console.log(y);
```



if (condition) {
  y = exp1
} else {
  y = exp2
}

# Ternary Operator


->

```
let y = ((condition) ? (exp1) : (exp2));
```

```
1   let y = ((10 > 5) ? (10) : (7));
2   console.log(y);
3
4   let x = ((10 < 5) ? (10) : (7));
5   console.log(x);
6
7   let a = ((true) ? (2 + 3) : (2 - 3));
8   console.log(a);
9
10  let c = 10;
11  let b = ((3 < 1) ? (2 - 3) : (++c));
12  console.log(b);
```

Errors | Warnings

```
» ▸ let y = ((10
  console.log(
  let x = ((10
  console.log(
10
7
5
11
← undefined
```

# Switch case

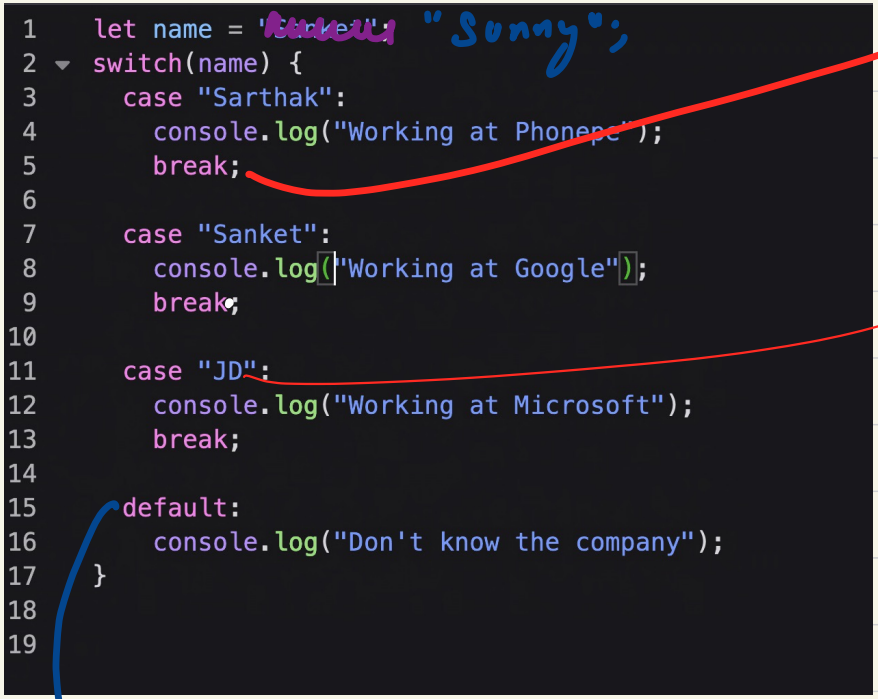
switch is a keyword

```
switch (value or expression) {

    case value 1 :
        // logic
        break;
    case value 2 :
        // logic
        break
    case value 3 :
        // logic
        what
    default.

}
```

break is a keyword

Switch (value / expression)

↳ whatever value or expression gets calculated, we can use that value & check for the case.




if we don't put break, then whatever case we hit, below that everything gets executed, till we find a break

if we use break within a switch, the moment we hit the break statement, we exit out of the switch.

default is not mandatory → if we don't have default, & none of the case matches, we don't do anything.

Q. Given 3 no. , a , b, c , which represent coefficient of a quadratic eq^n.

$ax^2 + bx + c$.

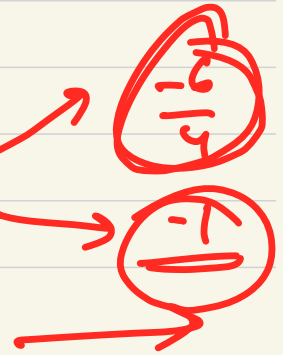
(assum real roots)

Find the roots of the quadratic eq^n:

$b^2 > 4ac$

$2x^2 + 5x + 3$

$$\rightarrow \frac{-5 \pm \sqrt{25 - 24}}{4} \rightarrow \frac{-5 \pm 1}{4}$$



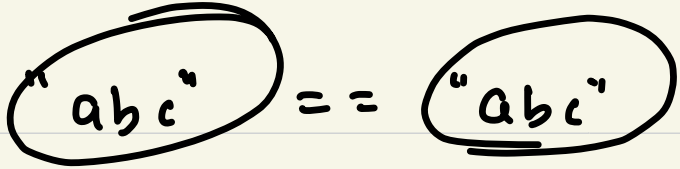
Hint: → Math.sqrt( x );

$$ax^2 + bx + c$$

$$\frac{-b + \sqrt{b^2 - 4ac}}{2a}$$

$$\frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

Math.sqrt(x) → $\sqrt{x}$


"abc" == "abc"