* Variable naming conventions / rules :

1. Variable names cannot start with a number.
2. It can only have letters, underscores, numbers, dollar.
3. you cannot use a reserved keyword as a variable name.
   (let, typeof, function, const, var, ...)

good practices

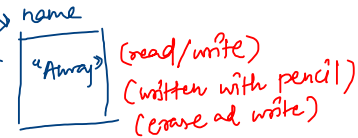
4. CamelCase
   my first job ⇒ myFirstJob
   current year ⇒ currentYear
   is javascript fun ⇒ isJavascriptFun

5. for constant values, all uppercase

* 6. variable names should be descriptive/meaningful, (v. imp cleaner code)
   (It should tell what value it is holding)

★ how to declare a variable ?

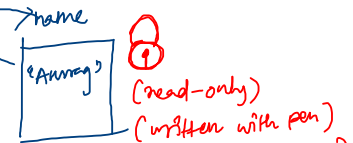
✓ ① `let name = "Anurag";`
`console.log(name)`



(read/write)
(written with pencil)
(erase and write)

1st preference should be const.

✓ ② `const name = "Anurag";`
`console.log(name);`
`X name = "priya"`
(const variables cannot be changed)

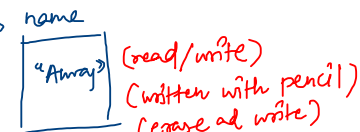


(read-only)
(written with pen)
(you cannot erase)

`const lastJob;`
`console.log(lastJob);` } cannot have empty

(var, let are a bit different (function, scope) next week)

~~③~~ `var name = "Anurag";`
`console.log(name);`
(not used in modern JS)



(read/write)
(written with pencil)
(erase and write)

* operators :

1. parseInt(132.862) ⇒ 132

2. exponential operator (**) ⇒ a ** b ⇒ a^b

   Eg: 2 ** 3 ⇒ $2^3 = 8$

   16 ** 0.5 ⇒ $16^{0.5} = 16^{1/2} = \sqrt{16} = 4$

   27 ** 1/3 ⇒ $27^{1/3} = \sqrt[3]{27} = 3$

3. modulo operator (%) ⇒ 11 % 3 = 2
   (remainder) ⇒ 19 % 4 = 3

   a % b ⇒ remainder of a/b

4. addition also works with strings
   (Join / concatenate)

   s1 = "Ranbir"
   s2 = "kapoor"
   s3 = s1 + s2 = "Ranbir" + "kapoor"
   = "Rabirkapoor"

$$3\overline{)11}\quad -9 \quad (2)$$

$$4\overline{)19}\quad -16 \quad (3)$$

5. \n is a special character which takes the control to print from next line.

== vs ===

1. `1 == 1` — It checks only **value** → value = 1, value = 1 → true

2. `1 === 1` — It checks **value** and **datatype** → value = 1, dtype = Number; value = 1, dtype = Number → true

3. `"1" === 1` — value = 1, dtype = String; value = 1, dtype = Number → false

4. `"1" == 1` — value = 1, value = 1 → true

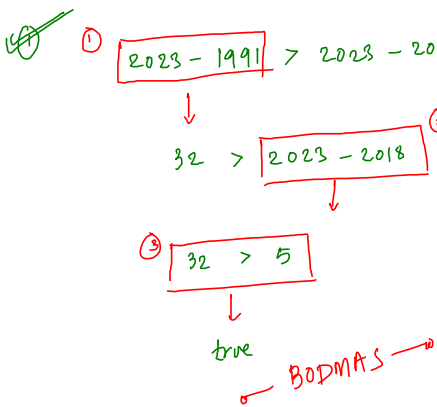
- (==) loose checking
- (===) strict checking

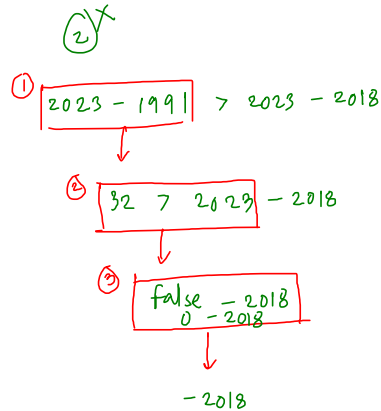
- (!=) loose
- (!==) strict

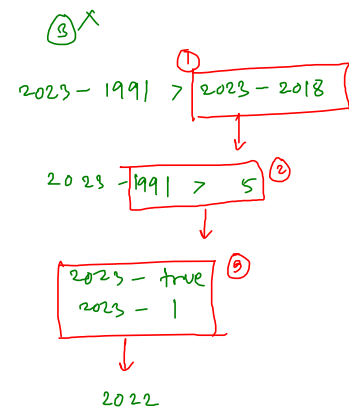
★ operator precedence :

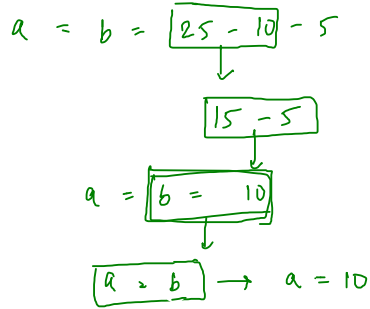
```
const curYear = 2023
curYear - 1991 > curYear - 2018
```

true - 1
false - 0

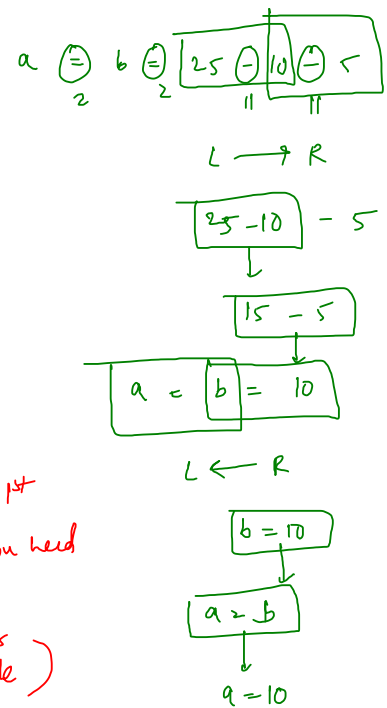









```
console.log(a,b)
10 10
```



* precedence - which one to evaluate 1st

associativity - direction in which you need to evaluate.

(BODMAS intuition is good for us without looking at the table)