**1.Explain the difference between null and undefined in javascript.**

null: is an assignment value that represents the intentional absence of any object value

Undefined: means that a variable has been declared but has not yet been assigned a value to

It is a default value for uninitialized variables , function arguments that are not provided.

**2.**



**console.log(‘10’+5) → 105**

* Operator in javascript can be used for both addition and concatenation.

Because + operator encountered a string ‘10’ it converts 5 into a string and concatenates

**console.log(‘10’-5) →5**

* Operator only performs subtraction.

**console.log(true+2) →3**

The boolean value true can be represented as 1. So js considers true as 1 and performs addition with 2 which outputs 3

**console.log(false+undefined) →NaN**

The boolean value false can be represented as 0 . Undefined is not a number and adding 0 to that would also be not a number

**3. What is the difference between == and === in JavaScript? Provide examples**

‘==’ compares two values without comparing their types.(converts one data type to another)

Ex: console.log(5==’5’)→ true

console,log(‘1’==true)→true

‘===’ compares both type and values of the operands

Ex: console.log(5==’5’)→false

console.log(‘1’==true)→false

**4.**



**console.log(0==false)→true**

== internally converts one data type to another . false can be represented as 0.

**console.log(0===false)→false**

=== compares the type and value of the operands

**console.log(‘ ’==0)→ true**

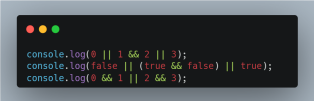
The string ‘ ‘ (a single space) is first converted to a number. When a string is compared with a number, JavaScript tries to convert the string to a number.

The string ‘ ‘ is converted to 0

**console.log(‘ ’===0)→ false**

The ‘===’ operator does not convert a datatype to another. It just compares whether the operands are of same data type and have the same value or not

5.



**console.log(0| | 1&&2 | |3))→ 2**

* 1&&2 → 2
* 0 || 2 || 3 → 2
* 0 || 2 would return 2 and javascript stops the further evaluation

**console.log(false || (true && false) || true)→true**

&& has higher precedence than ||

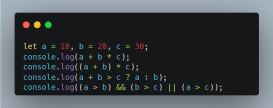
* true && false would return false. Then the expression would be→ false || false || true
* false || false would return false
* false || true would return true

**console.log(0 && 1 || 2 && 3 )→3**

&& has higher precedence than ||

* 0&&1→ evaluates to 0
* 2&&3→ evaluates to 3
* 0 || 3 → evaluates to 3

6.



**console.log(a+b\*c)**

Multiplication takes precedence. b\*c, 20\*30 =600

Then, 10+600=610

**console.log((a+b)\*c)**

(a+b) takes precedence, 10+20=30

Now, 30\*30=900

**console.log(a+b > c ? a: b)**

Ternary operator:

a+b=10+20=30, 30>30 is false

So, It returns a i.e 20

**console.log((a>b) && (b>c) || (a>c))**

(a>b) : 10> 20 = false

(b>c) : 20>30 = false

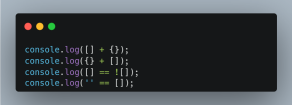
(a>c) : 10>30 = false

(a>b) && (b>c): false && false=false

false || (a>c)= false || false=false

Therefore, output is false

7.



**console.log([ ]+{ })**

When + operator is used with arrays and objects, javascript converts the operands into primitive values

[ ] is converted to empty string “ ”

{} is converted to a string representation. `”[object Object]”`

The result of concatenating empty string with `”[object Object]”` is `”[object Object]”`

**console.log({ }+[ ])**

{} is treated as an empty block

+[ ] is treated as a string operation and this results in empty array converting to [object Object]

**console.log([] == ![])**

![] results in false, which converts to 0

Javascript converts [] to empty string `” ”` which converts to 0

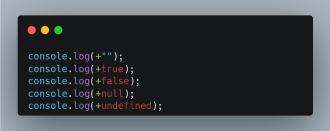
0==0, will return true

**console.log(‘ ‘== [])**

‘ ’ is an empty string

[] is converted into empty string

Therefore, empty string==empty string which results in true

8.

**console.log(+””)**

The + operator converts the string into a number i.e into 0. Therefore the result is 0

**console.log(+true)**

The + operator converts the boolean value true into number 1. The result is 1

**console.log(+false)**

The + operator converts the boolean value false into number 0. The result is 0

**console.log(+null)**

The + operator converts the null into number 0.The result is 0.

**console.log(+undefined)**

The + operator cannot convert undefined into a meaningful number. Therefore in results in NaN