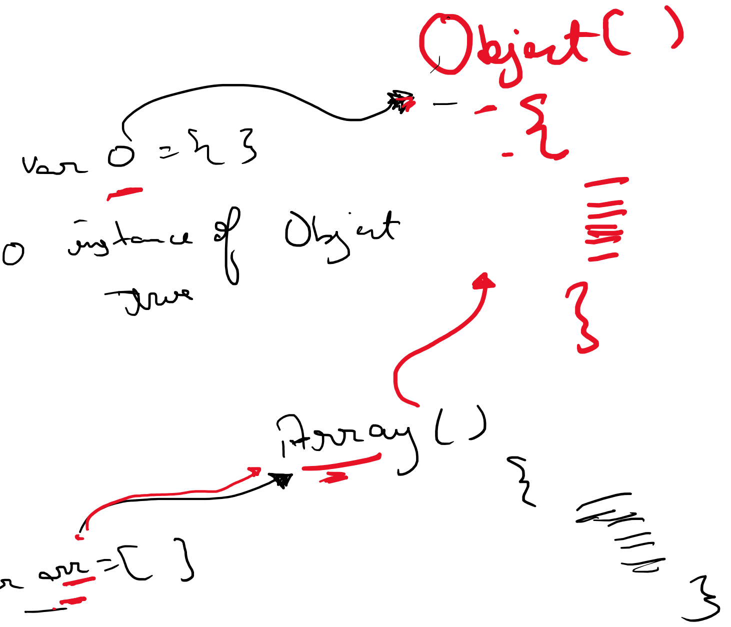
Top level

GOD level

arr instance of Object

true



o instance of Object

true

arr instanceof Array

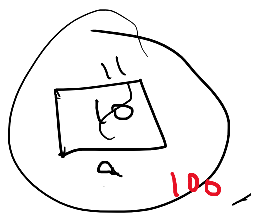
true

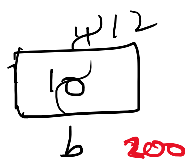
IMMUTABLE TYPES

value copy

var a = 10

var b = a;





distinct memory address creation

a++

11

b++

b++

12

primitives / Immutable types → copy → 2 different values

changes done in Either original or copied value, does not affect the other.

Mutable types

array literal

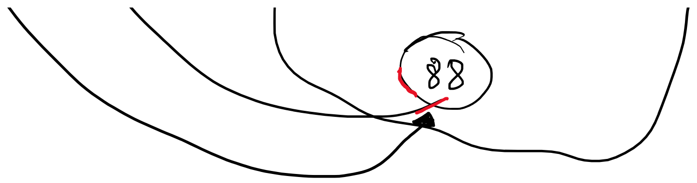
Object literal

var arr1 = [ ]

var arr2 = arr1

changes get

values get ~~[illegible]~~ mutated



→ changes done in either original value or copied value will ultimately change the other also.

$\boxed{\text{Mutability}}$.

$\boxed{\text{primitive value}}$

var $\boxed{\text{obj1}}$ = { _:_ , _:_ , }

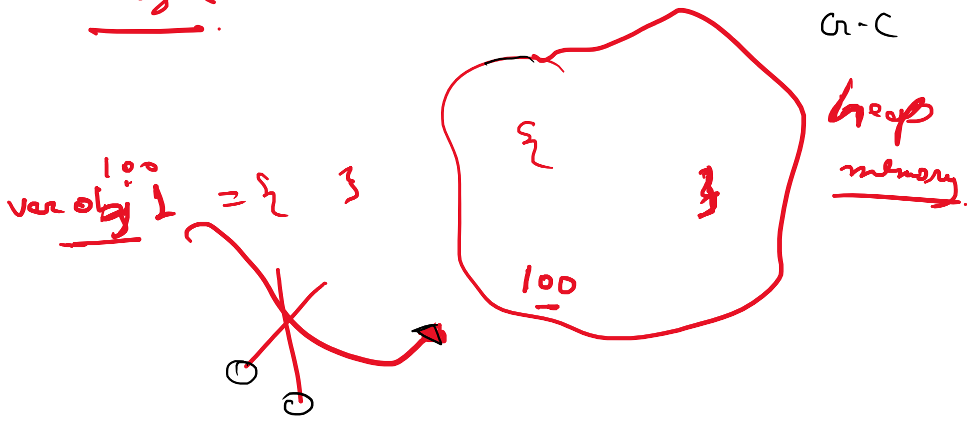
garbage collection

G-C

V8 engine

type same

value empty.



obj1 → empty object

$\boxed{\text{null}}$

is like a scissor

Primitive Values

refer address

a. toUpperCase()

(.) operator

var a = "Priya";

'priya'

'PRIYA'

var surname = new String("...")

object type

var a = new String(" ")

a. toUpperCase();

value type

Powers

a. toUpperCase( ); value type

'PRIYA' ;

normal human.

Powers

a.

AUTOBOXING

a;

'priya' ; value type

Primitive values → WRAPPER TYPE → Type of object

string → String

number → Number

boolean → Boolean   var bool = new Boolean ( )

bigInt → BigInt

symbol → Symbol ( )

null→empty value , undefined → empty value