20-08 Advanced Javascript Concept - 1 2 3

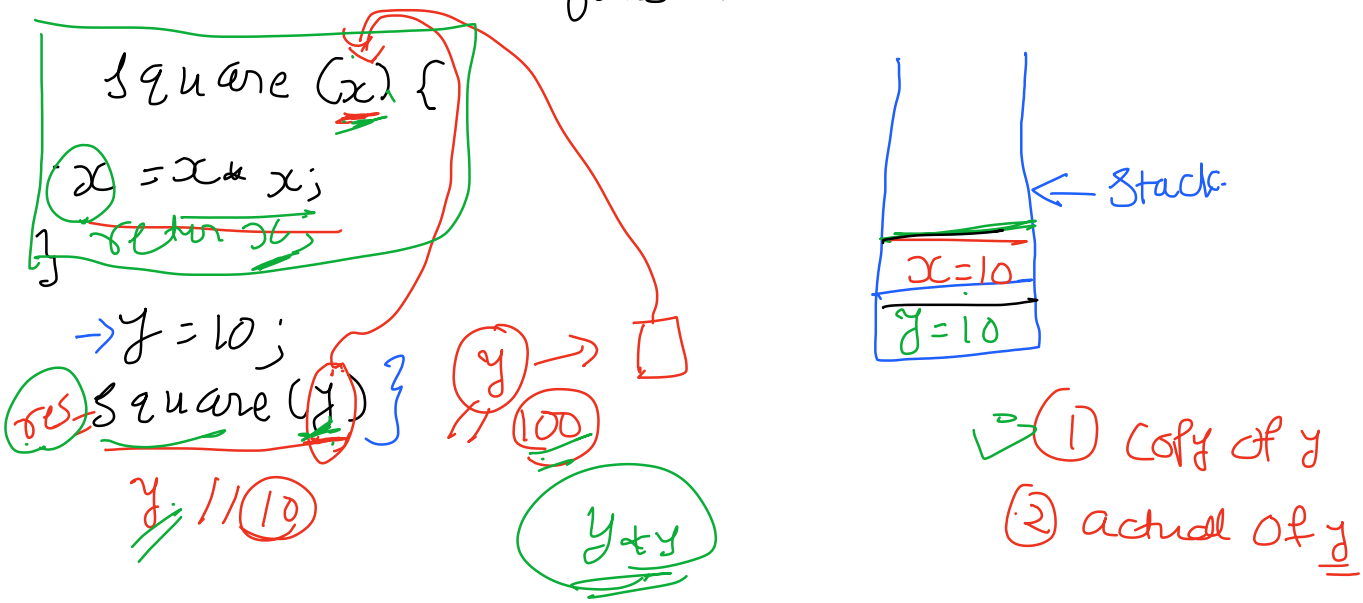
Saturday, 20 August 2022 10:13 AM

1. Pass by value & Pass by reference.
2. Pure & Impure functions.
3. Closures.

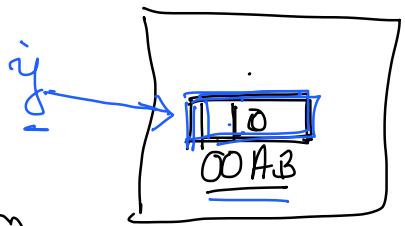
In JS, all fn arguments are Pass by value.

↓

copy the value of variables into fn arguments

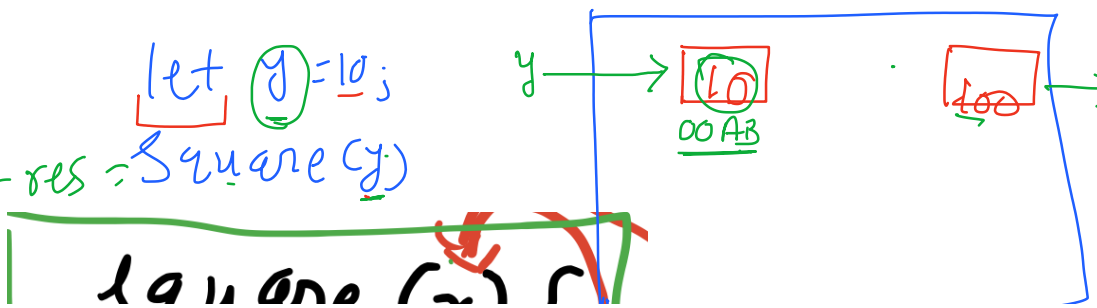


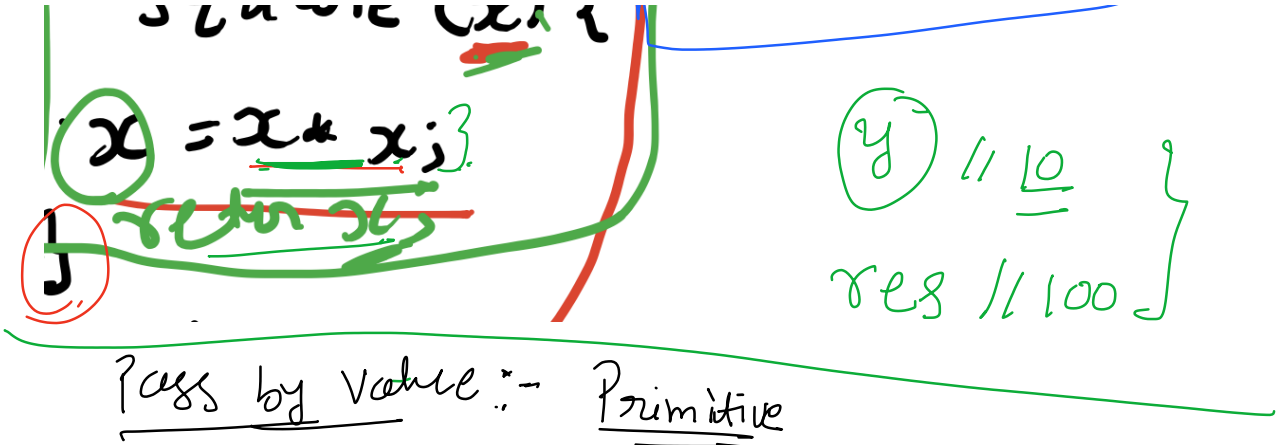
let y = 10;



Console.log(y)

let y = 10;
let res = square (y)



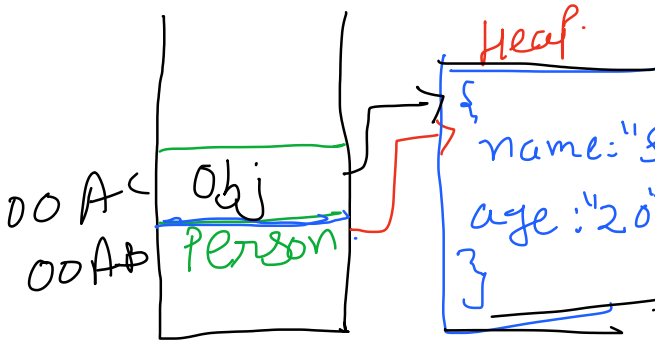


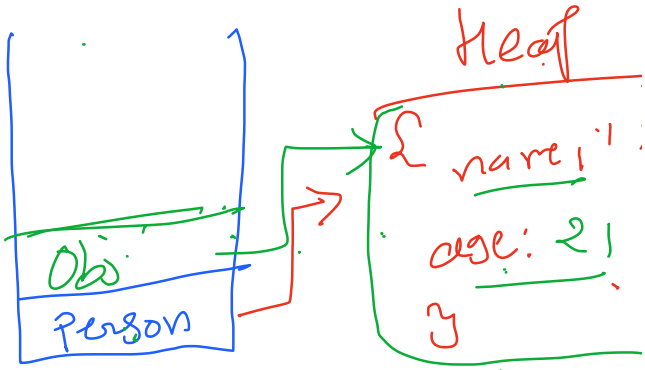
Pass by Value :- Primitive

1. numbers
2. String
3. Boolean

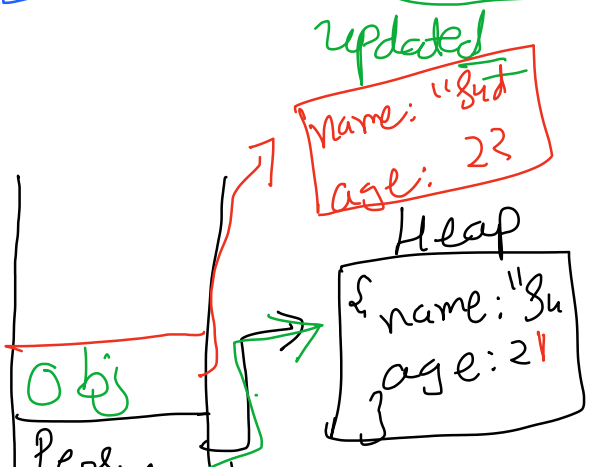
Object, Arrays } Pass by ref

```
let person = {
  name : "Sudhakar",
  age : 20
}

function increaseAge(obj){
  obj.age+= 1;
}

increaseAge(person);
console.log(person);
```



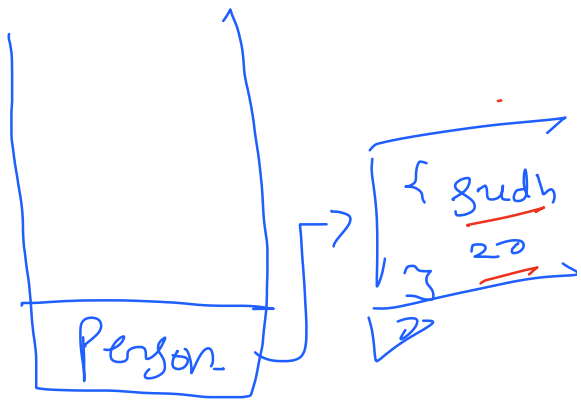



```
1  let person = {
2    name : "Sudhakar",
3    age : 20
4  }
5
6  function increaseAge(obj){
7    obj.age+= 1;
8    obj = {
9      name : "Sudhanshu",
```



```
    age : 22
};
} Con.log(obj);

increaseAge(person);
console.log(person);
```

```
let person = {
  name : "Sudhakar",
  age : 20
}

function increaseAge(obj){
  obj = {
    name : "Sudhanshu",
    age : 22
  };
  obj.age+= 1;//
  console.log(obj)
}

increaseAge(person);
console.log(person);
```

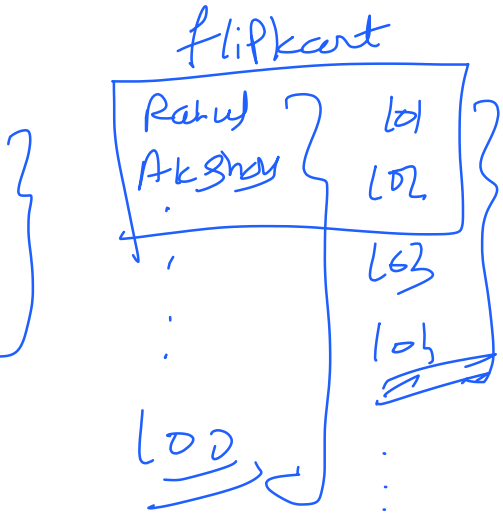


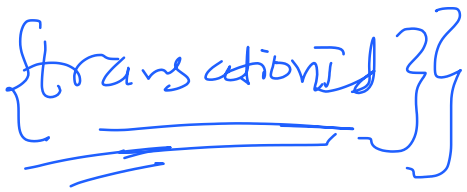
⑥ Pure fn & Impure fn

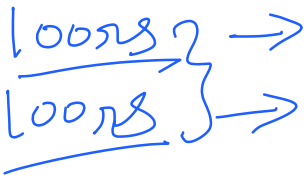
1. Pure
   1. Predictible
   2. Has no side effects
2. Impure
   1. unpredictible.
   2. Has side effects

```
function add1(x,y){
  let rand = Math.random() * 10;
  return x+y+rand;
}

console.log(add1(2,2))
// Math.random() generate random
number between (0,1]
// if i multiply with 10 => generate
no between 1 to 10
```


```
function add(x, y) {
    return x + y;
}

console.log(add(2, 2));  // 4
```








```
var globalVar = 1; //

function add(x,y){
  globalVar=0;
  return x+y+globalVar;
}
console.log(add(2,3))



function printGlobalVar(){
  console.log(globalVar);
}
printGlobalVar()
```

```
let mutateNum = 0;

const impureFunction = (num) => {
    return (mutateNum+=num);
}

console.log(impureFunction(5)); //
console.log(impureFunction(5)); //
console.log(impureFunction(5)); //
console.log(impureFunction(5)); //
```

✓ Hoisting
✓ Closures
✓ Currying
} Imp

⑭ Closures :-

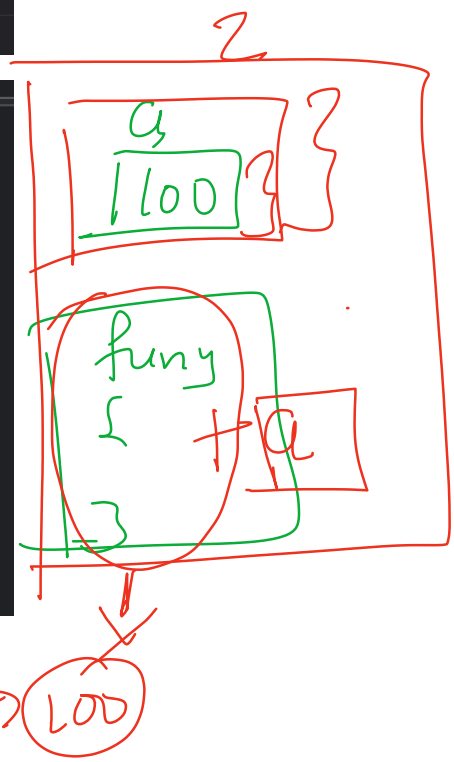
```
// Nested function

function greet(name){ // John
  function displayName(){
    console.log(`Hi ${name}`)
  }
  displayName()
}

greet("John");
```

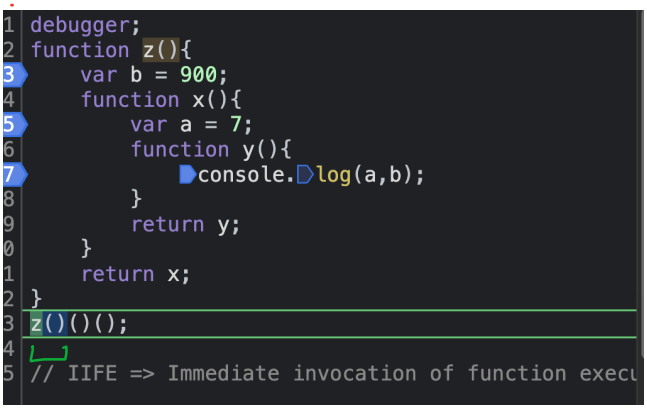
```
//Returning a funcition

function greet(name){ // John
  function displayName(){
    console.log(`Hi ${name}`)
  }
  return displayName;
}

let g1 = greet("John");
console.log(g1);
g1();
```

Errors | Warnings | Logs | Info | Debug | CSS | XHR

```
>> ▸ //Returning a funcition
   function greet(name){ // John
     function displayName(){
       console.log(`Hi ${name}`)…
▸ function displayName()        debugger eval code:11:9
Hi John                         debugger eval code:5:13
← undefined
```

```
> debugger;
  function x(){
    var a = 7;
    function y(){
      console.log(a);
    }
    a = 100;
    return y;
  }
  var z = x();
  console.log(z);
  //----------------------

  z();
```





```
debugger;
function z(){
    var b = 900;
    function x(){
        var a = 7;
        function y(){
            console.log(a,b);
        }
        return y;
    }
    return x;
}
z()()();

// IIFE => Immediate invocation of function execu
```