Day-22(mcp , cep , gec , callstack , hoisting)

Monday, 2 October 2023 7:49 PM

#

(Loops, cont - break, methods
let, var, const)

→ '2' + '2' - '2'

↓

'22' - '2' = 22 - 2 = 20

number

How actualy your code is being RUN.

Q What is min. line of code which you can write in JS.

↓

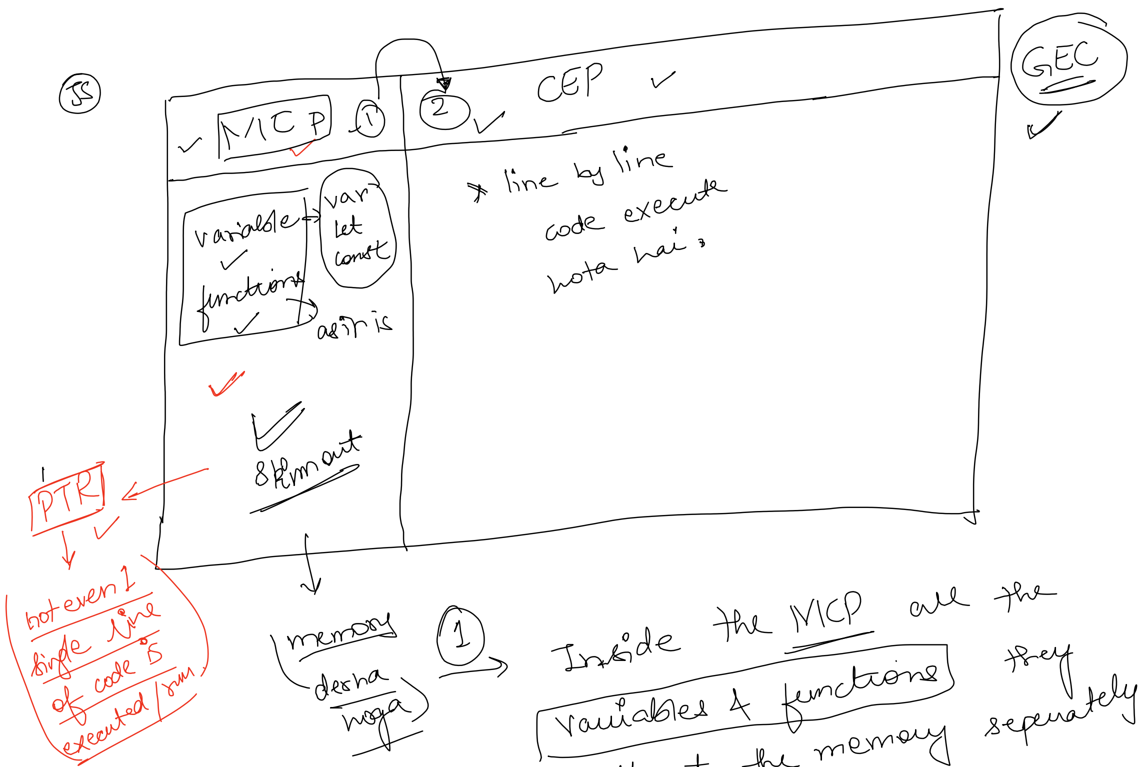
empty file

*** → Actualy your JS engine works

→ Whenever a JS code is run, global Execution Context created, and inside GEC we have ② diff. phases

① → Memory creation phase

② → Code execution phase


JS
MCP ①
② CEP
GEC
variable
functions
var
let
const
asir is
* line by line code execute hota hai.
skim out
PTR
not even 1 single line of code is executed / run
memory desha hoga

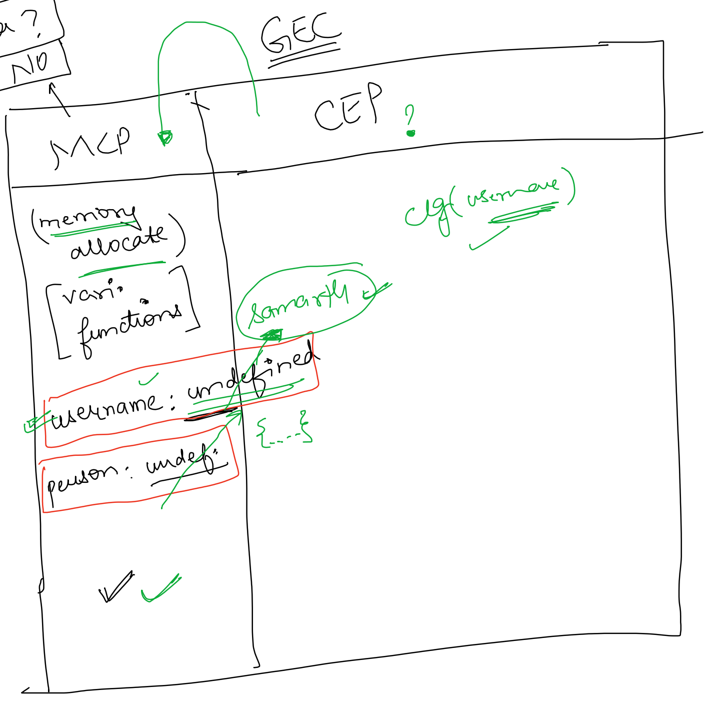
① → Inside the MCP all the variables & functions they all get the memory seperately in MCP.

② → CEP is where the code runs one after the other. line by line

Q. 16thi line chala? NO

```
var username = "samarth";
var person = {
    email : "sam@gmail.com",
    password: "samarth"
}

console.log(username);
```

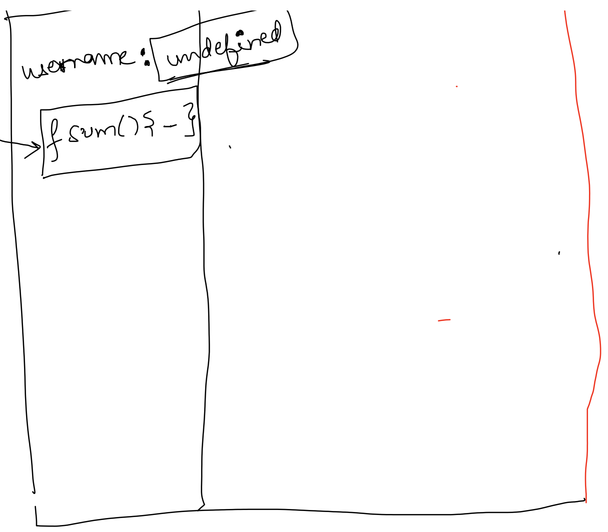


output

samarth

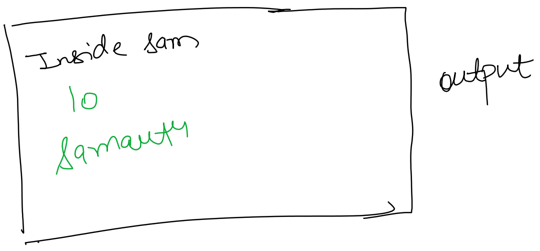
variable | function → skim out (Bahar)

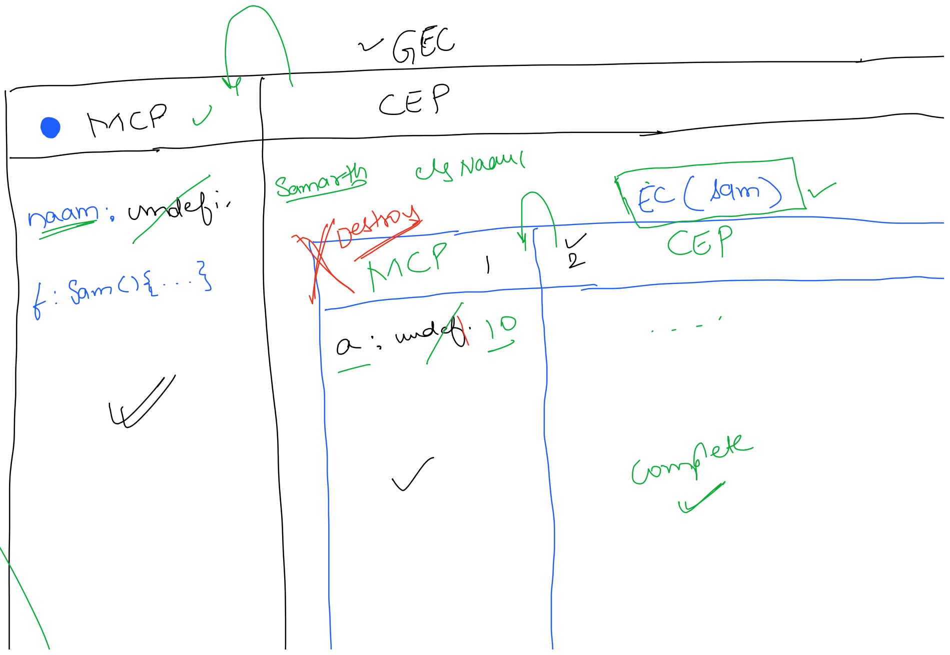
```
var username = "devi prasad";
function sum(){
    var num1 = 10;
    var num2 = 20;
    return num1+num2;
}
sum();
console.log(username)
```

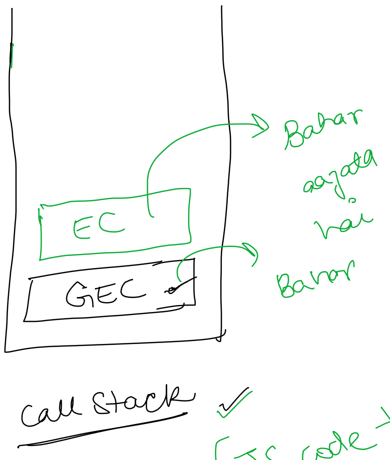


Whenever a func^n is called a new execution context is created with the name of the function.

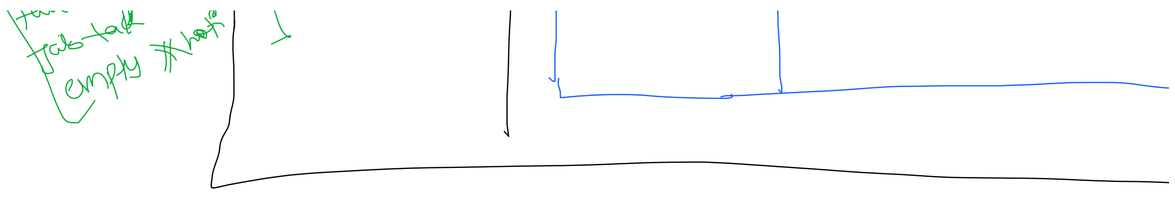
```
var naam = "samarth"
function sam(){
    console.log("inside sam");
    var a = 10;
    console.log(a);
}
sam(); (function calling)
console.log(naam);
```

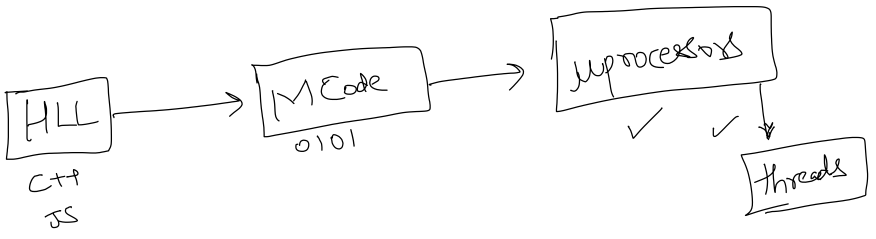


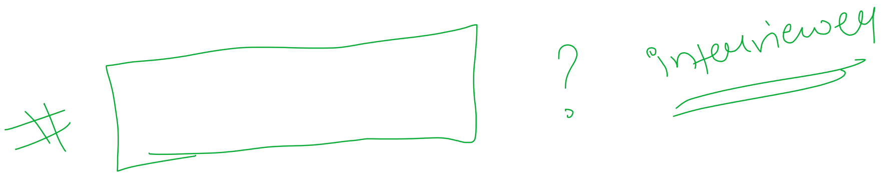


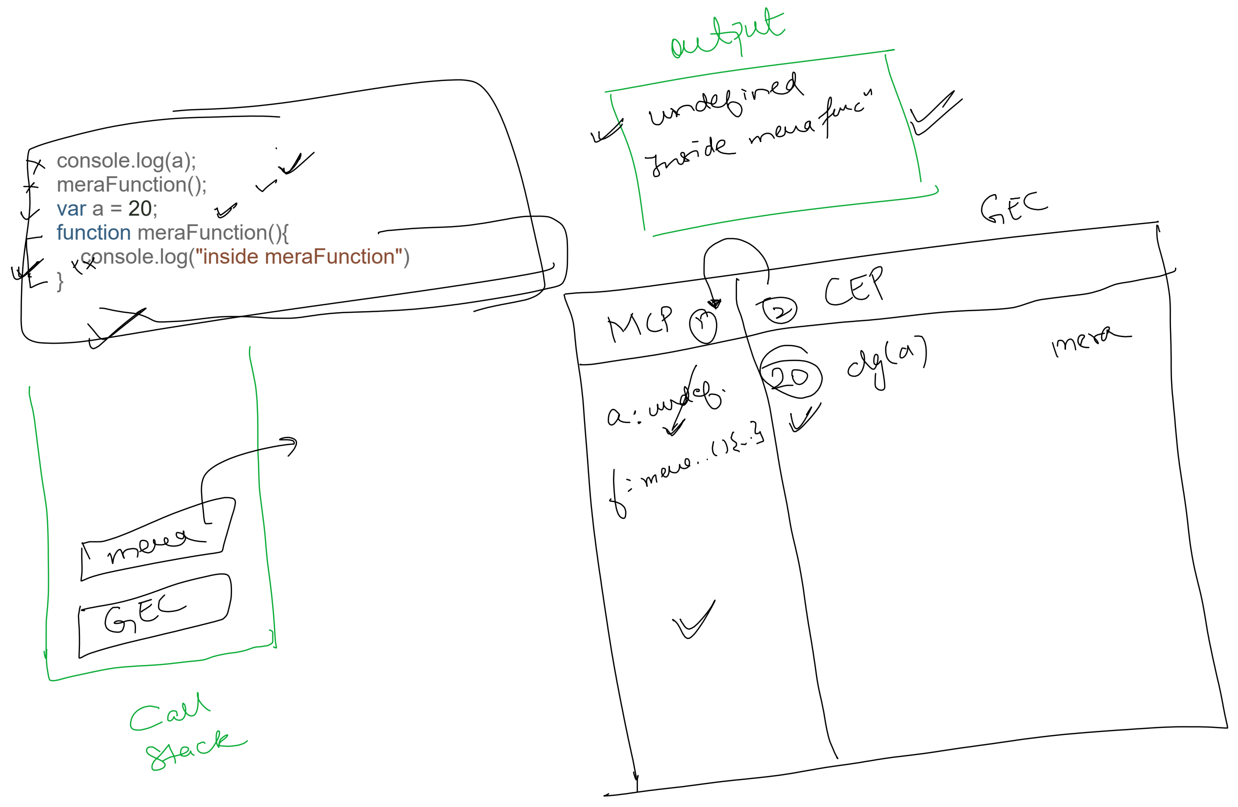


JS code tab tak chalta hai CS

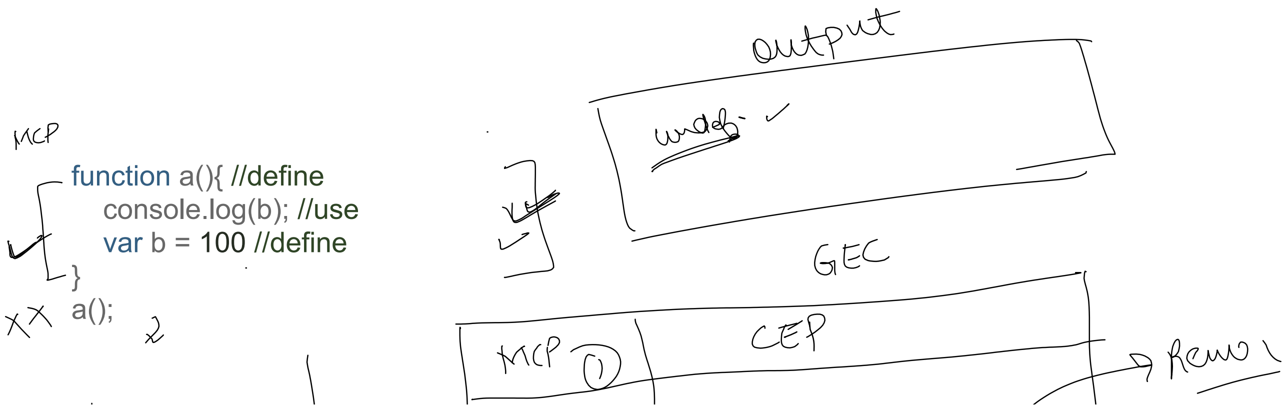




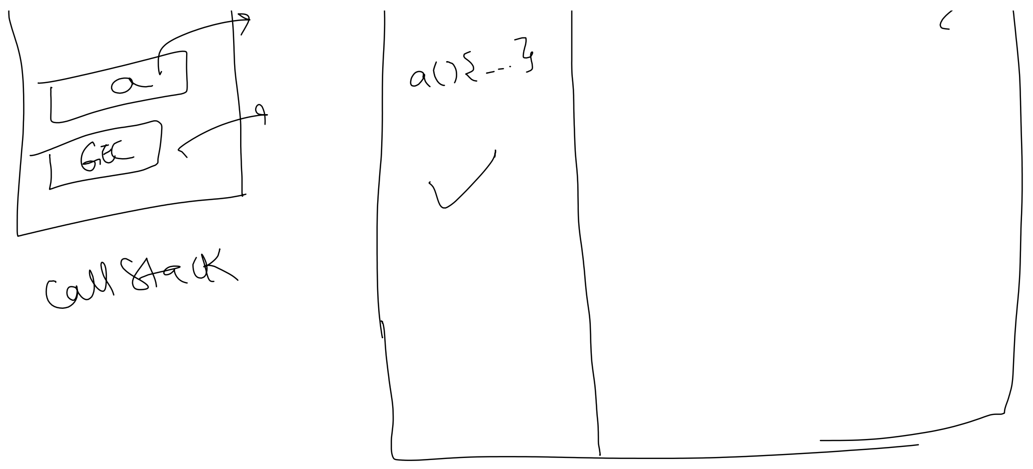




```
console.log(a);
meraFunction();
var a = 20;
function meraFunction(){
    console.log("inside meraFunction")
}
```



```
function a(){ //define
    console.log(b); //use
    var b = 100 //define
}
a();
```


a
GEC
Call Stack
a(){...}