Lecture 05 — Lecture 24:

From lecture 05 to lecture 24 I learned and did code on 'vscode'. so, there was no need to make notes on notepad or notebook manually.

Lecture 25:

Javascript Execution code + call stack.

⟹ javascript Execution content.

When we run javascript code then execution context is created.

$\boxed{\{\}}\xrightarrow{run}$ Global EC is created...

js-code (this) ~~is ct~~

jahan be EC create hota hai osko 1 variable refer krdia jata hai or wo variable yahan pr this hai.

- Ab her '1' ka execution content alag hota hai, like "browser" ka EC alag hota hai, nodejs,

Deno hr 1 ka alag hota hai.

jesy browser ka execution context window hota hai.

And everything is done in a thread. Also, js is a single threaded language.

There are different exeution content.

- Global Exeution context.
- function exeution context.
- Eval exeution context.

* javascript exeution code runs in two memory phases- $\boxed{\{\}}$ → js EC ode.

① Memory Creation Phase.

② Exeution Phase.

<u>NOTE</u>: Global exeution ko global environ-ment b bolty hain.

→ When js code run: EContext is created

<u>Steps</u>: 1 Global exeution (and allocate it to ~~its~~ (this variable).

2 Memory Phase

Val 1 کے اندر → undefined [illegible]

val 2 کے اندر بھی → undefined [illegible]

addnum کے اندر → function ki definition

resuet 1 → undefined

result2 → Undefined

codes

```
① let val1 = 10
② let val2 = 5
③ function addNum
        (num1, num2)
4 {   let total =
5        num1 + num2
        return total
6 }
let resuet1 = addNum
        (val1,
         val2)

let result2 =
    addNum (10, 2)
```

③ Execution phase:

→ i-val 1 $\overset{\text{کا اندر}}{\leftarrow}$ 10

→ ii- val 2 $\overset{\text{کے اندر}}{\leftarrow}$ 5

→ line iii to line vi tak kuch b execute nhi hoga.

→ line '7' pr 'addNum' function call horaha hai means invoke horaha hai tuo phir sy new Execution context create hoga.

So,

addNum → new variable environment + Execution thread.

ⓧ delete after making both phases it will delete (ⓧ)

↓

ab jitni dafa koi function invoke hoga utani dafa execution context create hoga and memory phase or execution phase bnygy. Global execution tuo 1 dafa bngaya hai wo nhi bnyga dubara. Dubara 2 phases exeute hongy.

ye jb dubara sy execution context banata hai, tu ye delete hojata hai.

ⓘ Memory phase

val1 = undefined

val2 = undefined

total = undefined

ⓘⓘ Execution Context (processing)

num1 → 10

num2 → 5

total → 15

~~ab~~ ye jo '5' line mai return hai ye kaha return hoga? ->ye basically parent executional context / global execution context hai osmai return hota hai.

So, I will automatically move to the step 3.

Execution phase.

where Val1 ← 10

Val2 ← 5

addNum → $\boxed{\text{new VE+ \& thread}}$ ← $\boxed{\text{Delete}}$

result1 = 15

result 2 = $\boxed{12}$ ← Here moved at the end.

line # '8' which is,

let result2 = addNum(10, 2)

↓

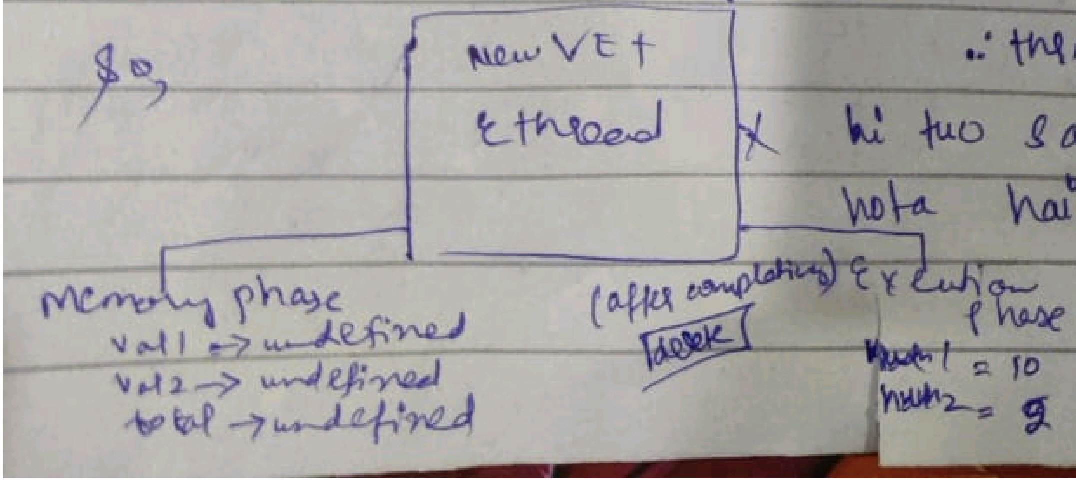
function is Invoking

So, again execution context is creating, once execution context is creating two phase run.

i- Memory creation phase

ii- Execution phase.

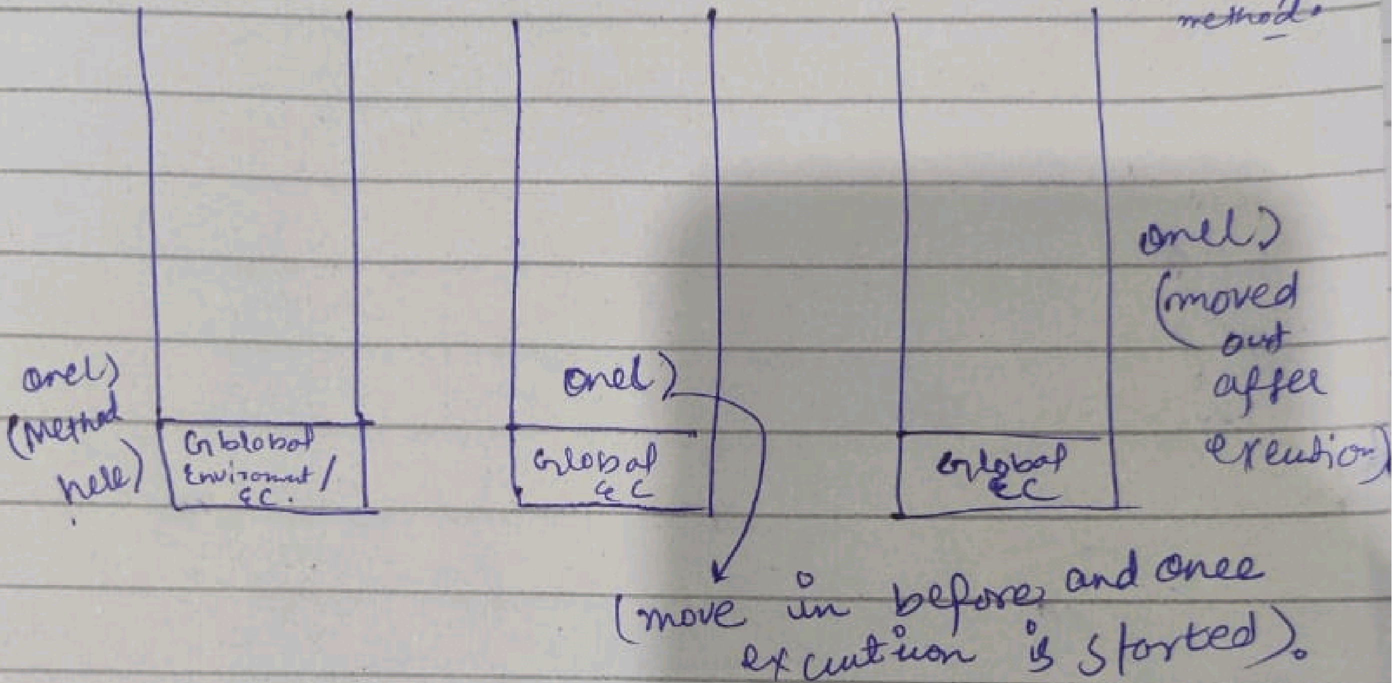
So,



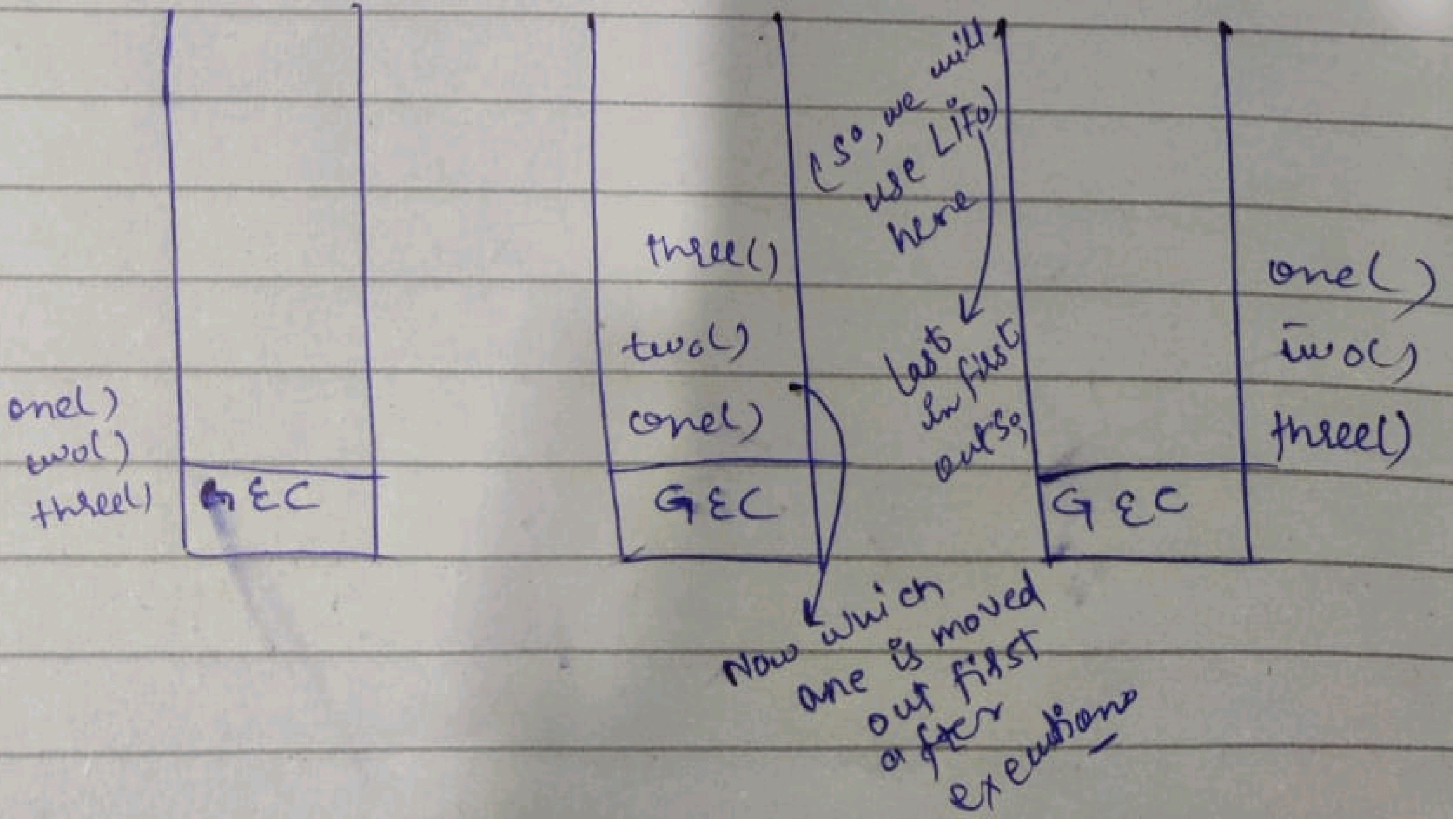
∴ thread per hi two sara execution hota hai js.

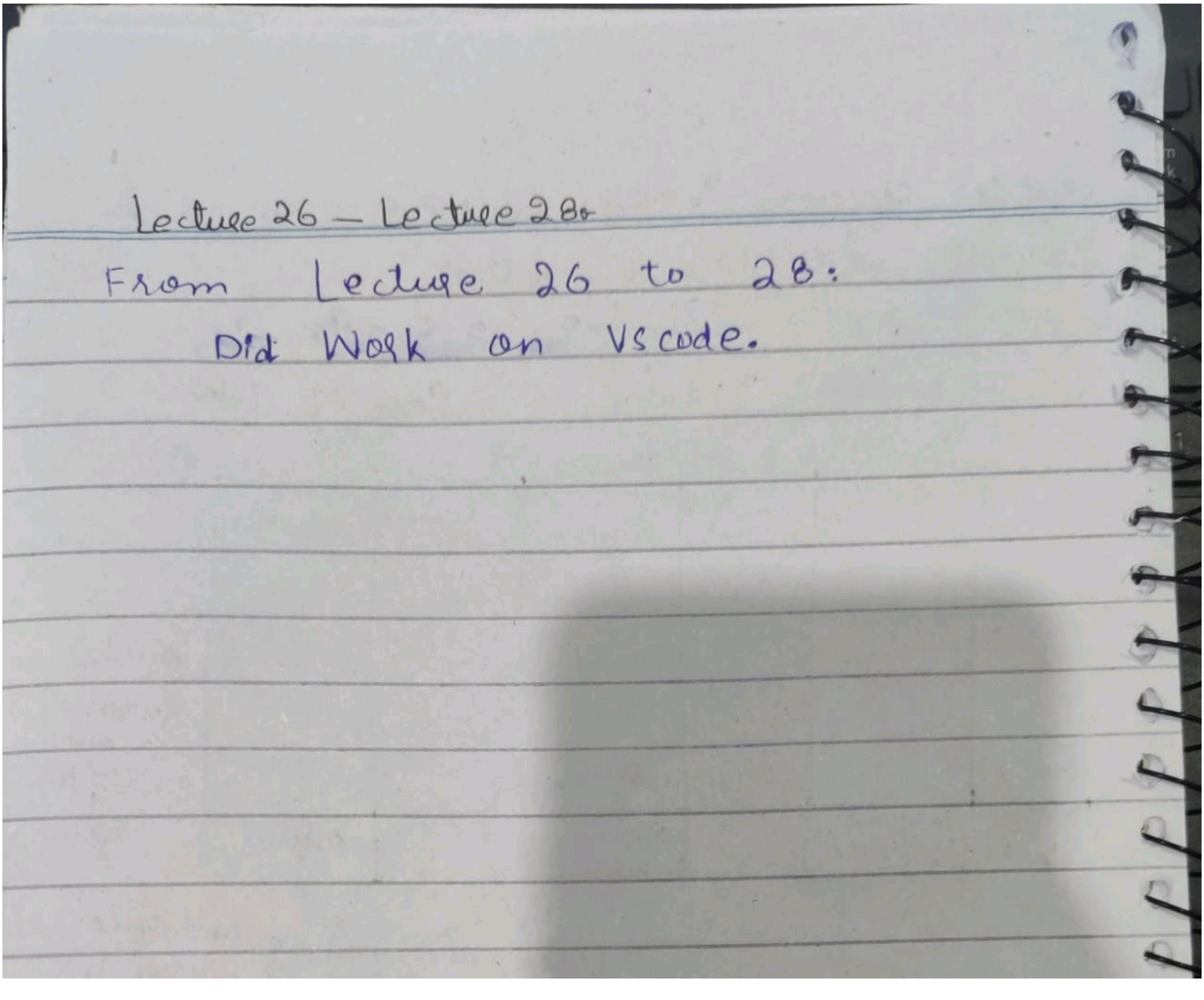
and then it moves to the exection phase and 12 will write these automatically.

→ Call Stack: Case 01 (when I have one function) method.


one() (Method here)
Global Environment / EC.
one()
Global EC
Global EC
one() (moved out after exection)
(move in before and once excution is started).

Case 02:- (when I have alot of functions and we call them together in GEC. like this:-





Lecture 26 – Lecture 28.

From Lecture 26 to 28:

Did Work on Vs code.