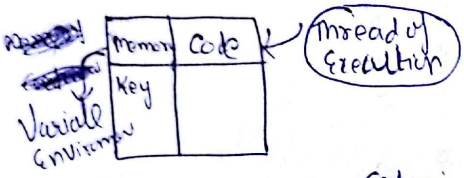
How Javascript works

① Everything happens in JS done in Execution context.

Execution context →



Memory component (Variable Environment) :- contains Variable & functions as key/value pair
Code component (Thread of Execution) :- Where whole JS code is executed.

- Javascript is synchronous, single threaded lang

Phases → ① memory creation → ② Code Execution Phase

```
1. var n = 2
2. function square(num){
3.     var ans = num * num;
       returns ans
4. }
5. var square2 = square(n);
6. var square4 = square(4);
```

① Memory Creation Phase :

Control moves to line 1 then ahead, takes memory & creates Execution Context first.

Variable Environment	Code
n : undefined	
square : {square code}	
square2 : undefined	
square4 : undefined	

Global Execution Context

② Code Execution Phase

In this phase at 1

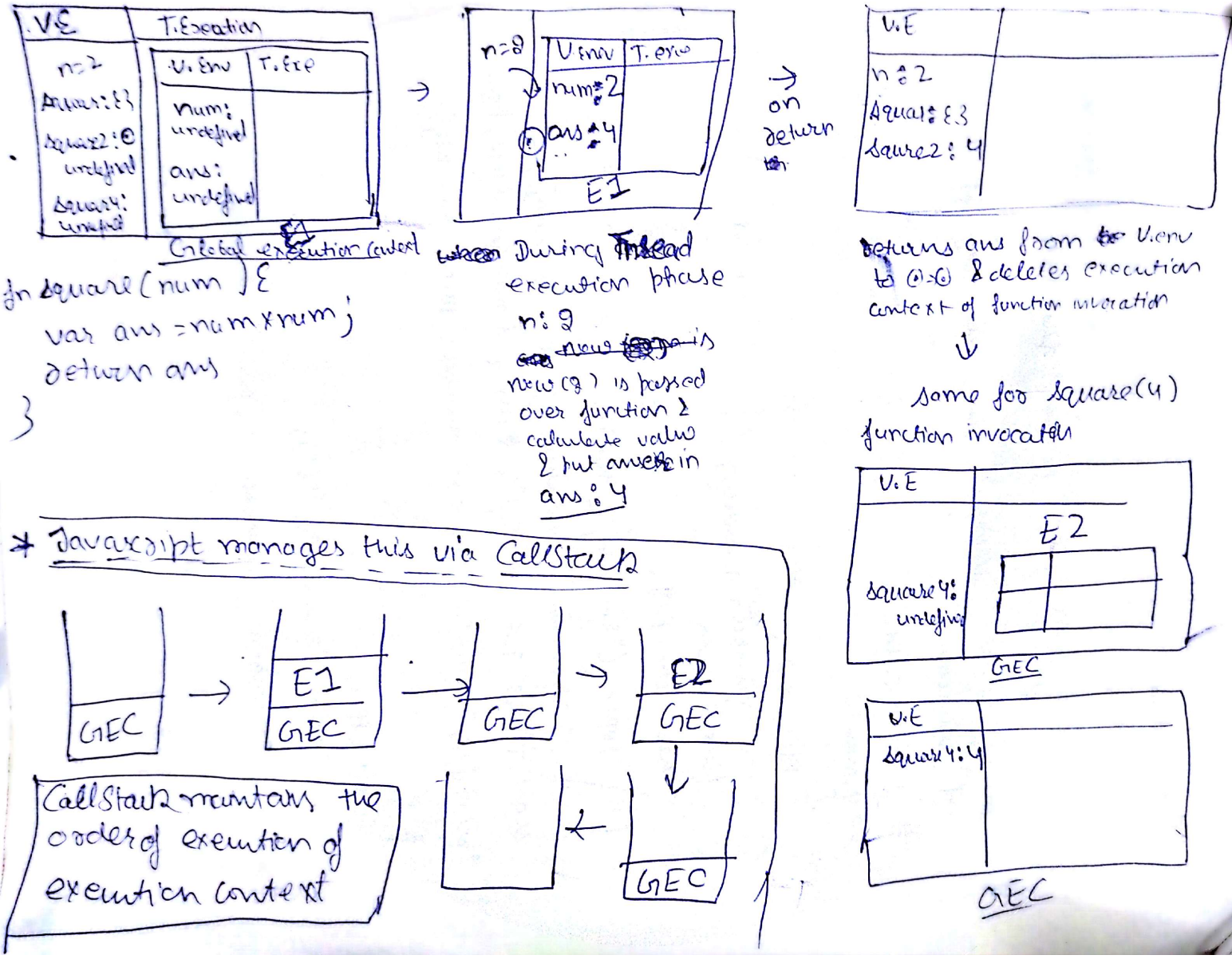
V.E	
n : 1	
square : {}	
square2 : {}	

nothing to with 2 to 4

on line 5 → function invocation

V.E	
n : 1	V.E | T.E
sq : {}	
sq4 : {}	

A new Execution Context forms E2



① What is Lexical env?

Lexical env is create with Execution context.

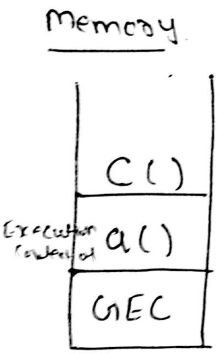
Lexical env = Local memory + lexical environment of it's parent.

② What is Scope chain?

```
Ex→
x = 10;
function a() {
  var b = 10;
  c();
  function c() {
    log(b);
    log(x);
  }
}
a();
```

Output →

10
20



→ c will search var b in it's Lexical environment, i.e first search for b in it's local & if not found then in it's parent scope.

Ex: Search x in local scope then it's ~~found~~ ~~scope~~ ~~not~~ search on lexical env of it's parent. ~~if not found~~

So ~~c()~~ c() in callstack has local scope & scope of it's parent. access to