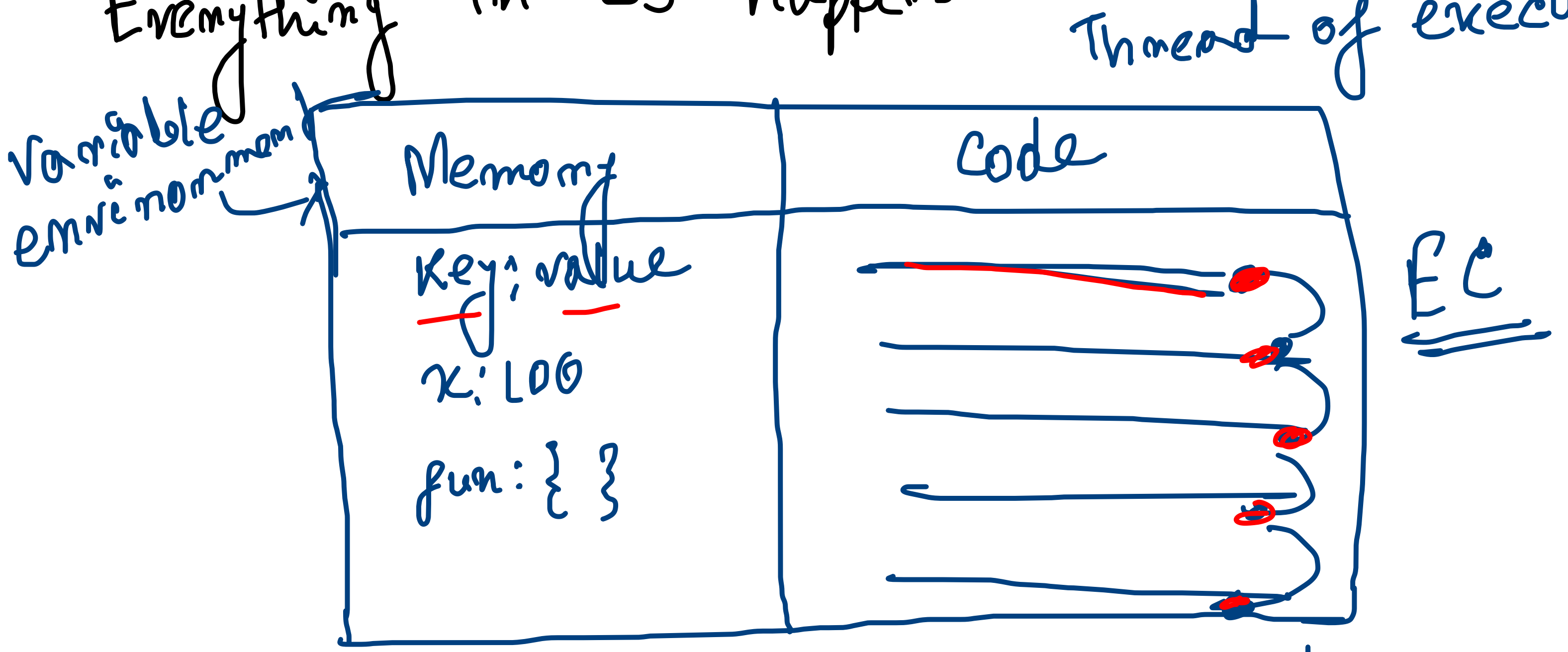
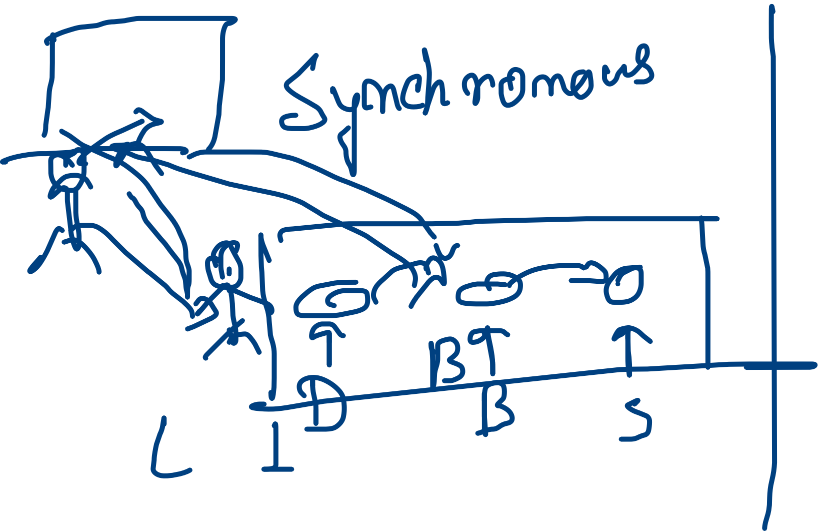
Execution Context?

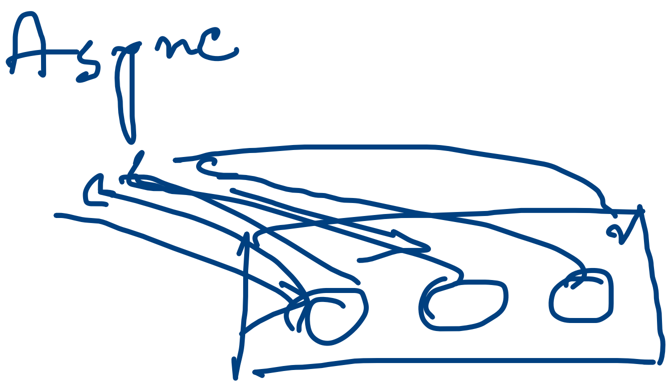
Everything in Js happens inside an execution context.



JS is a synchronous, single threaded language.

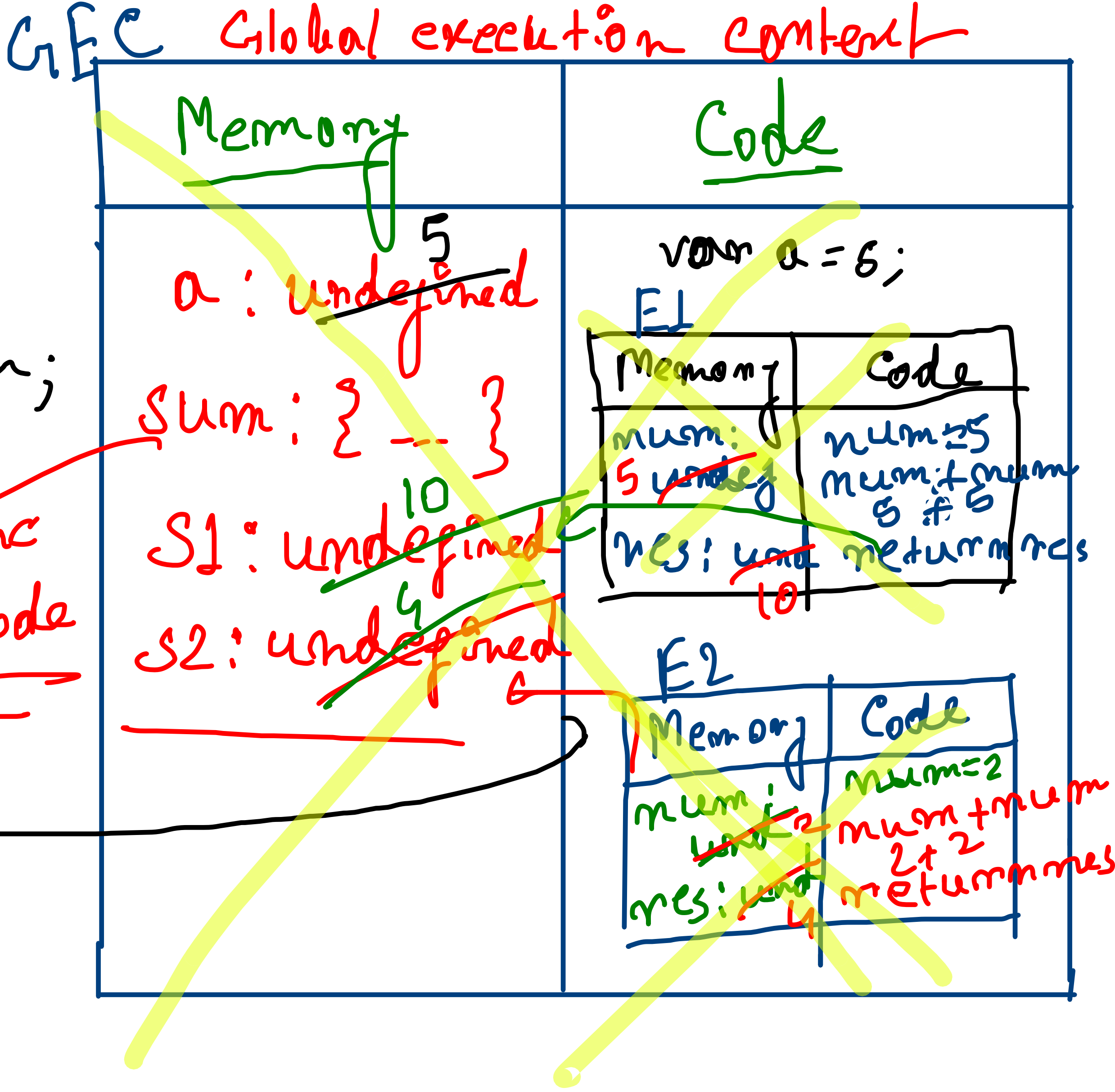
It execute one command at a time





```
var a = 5
function sum(num) {
      var res = num + num;
      return res;
}
var S1 = sum(a);
var S2 = sum(2);
```

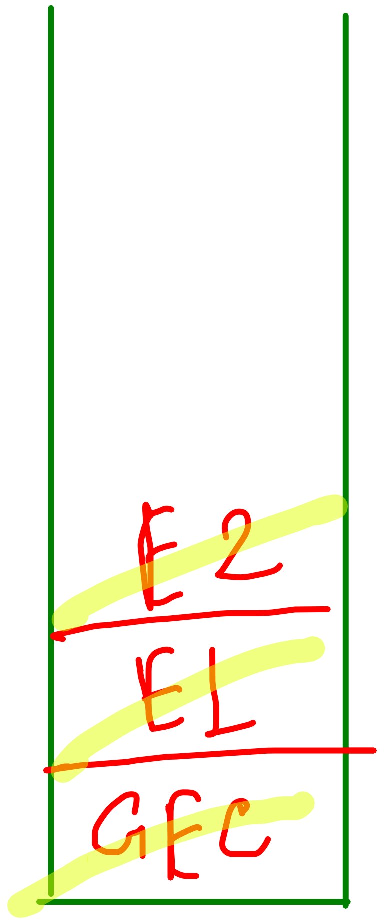
for func
total code



① Memory Creation

② Code execution

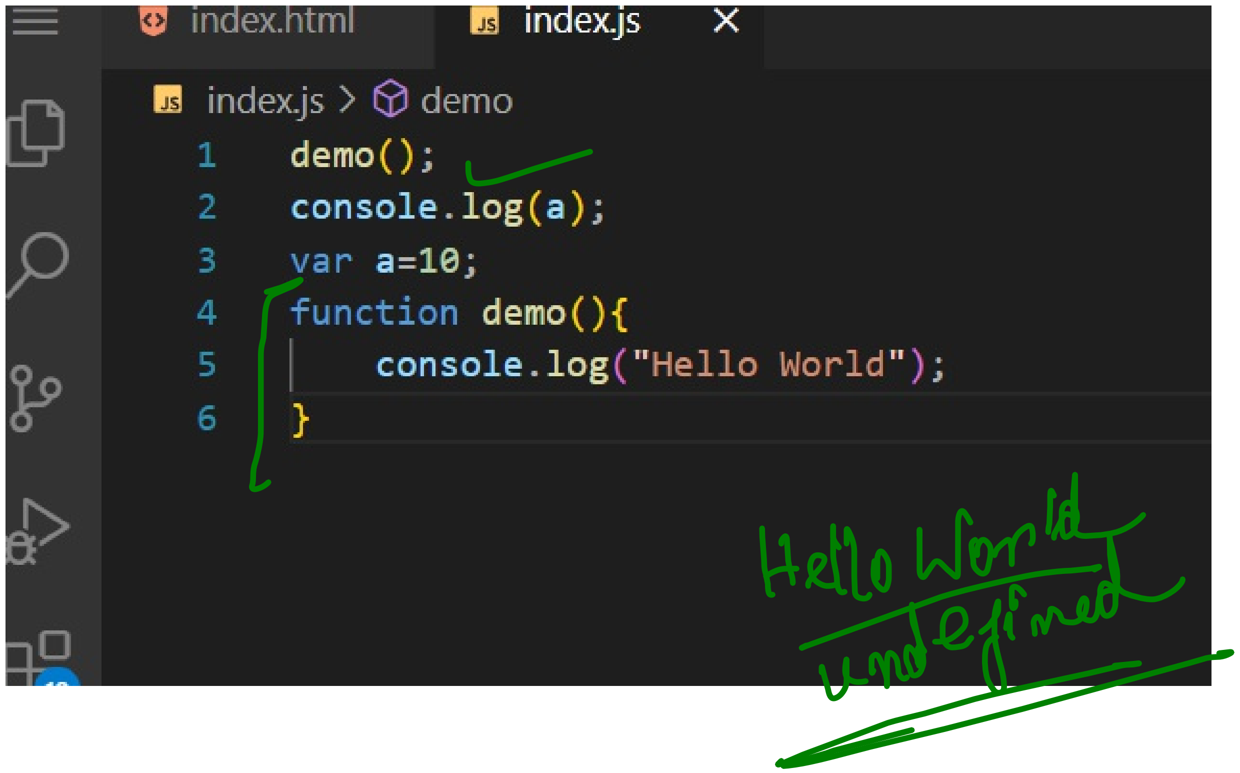
Call Stack



- Execution Context Stack
- Program Stack
- Control Stack
- Runtime Stack
- Machine Stack

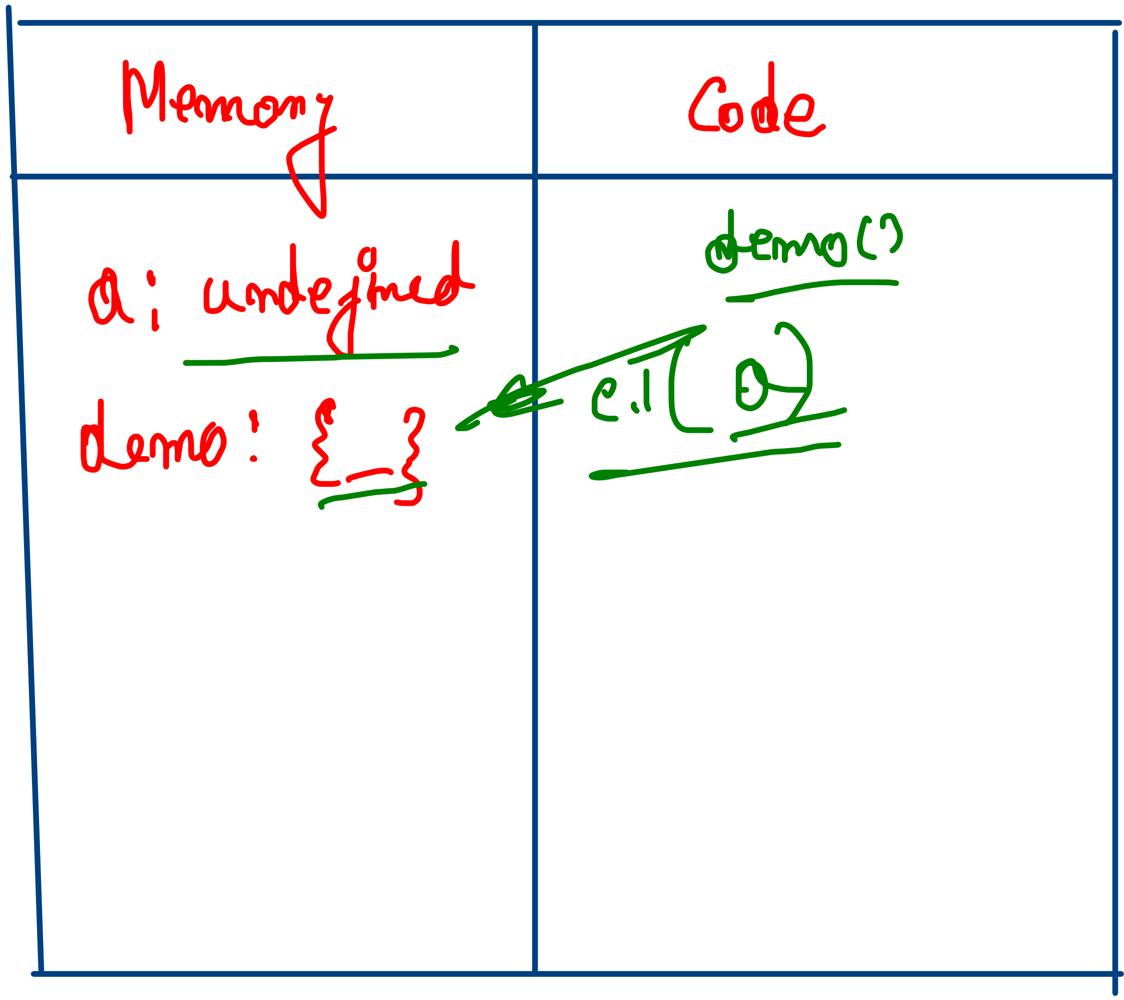
Callstack maintains the order of execution context

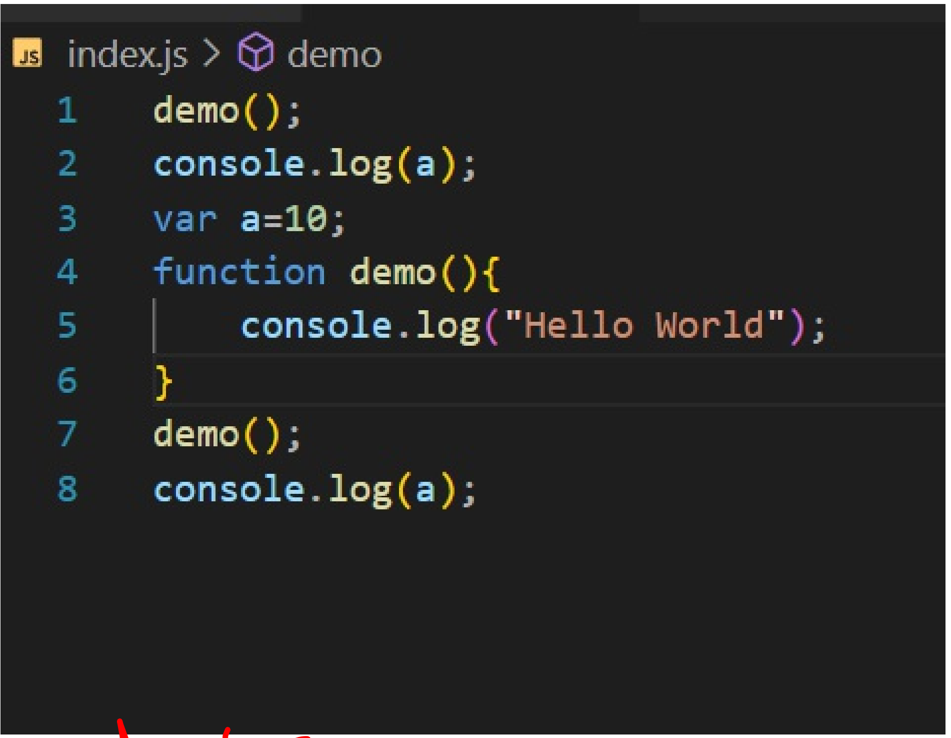
Hosting



```
demo();
console.log(a);
var a=10;
function demo(){
    console.log("Hello World");
}
```

Hello World
undefined

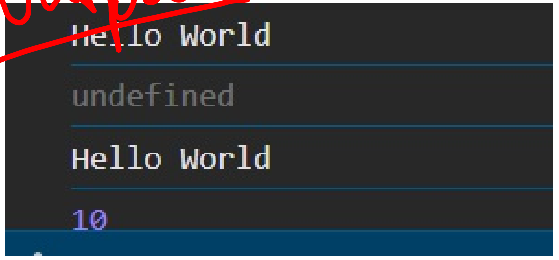




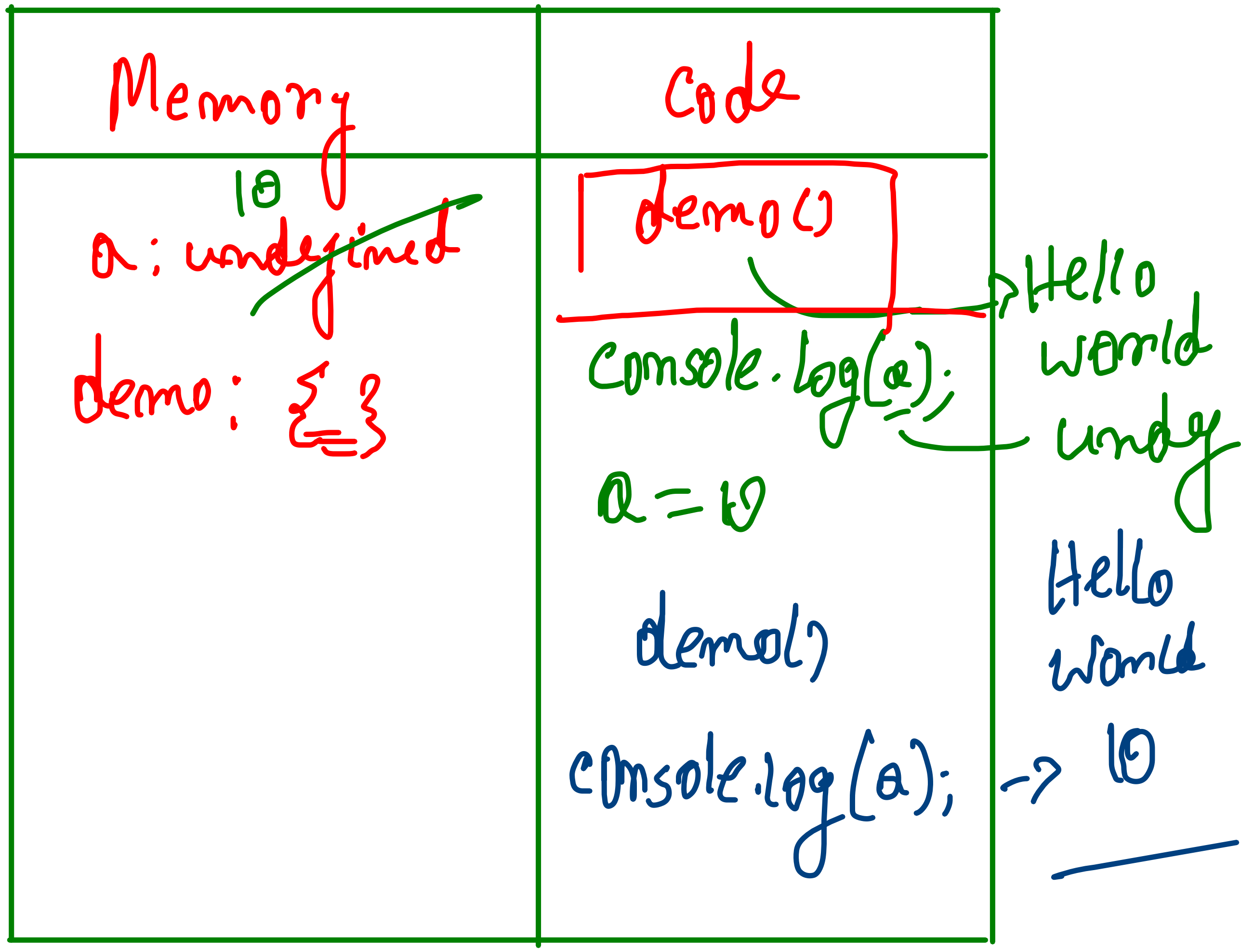
index.js > demo

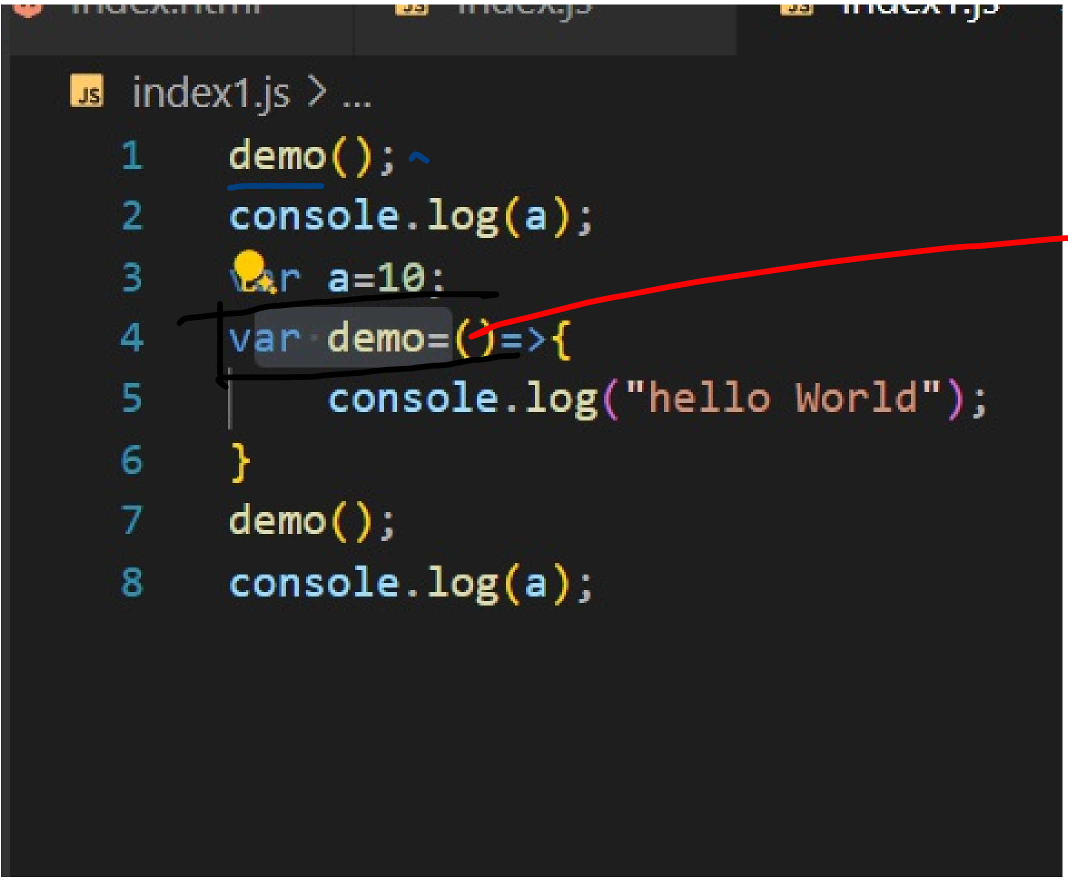
```js
demo();
console.log(a);
var a=10;
function demo(){
    console.log("Hello World");
}
demo();
console.log(a);
```

Output



```
Hello World
undefined
Hello World
10
```



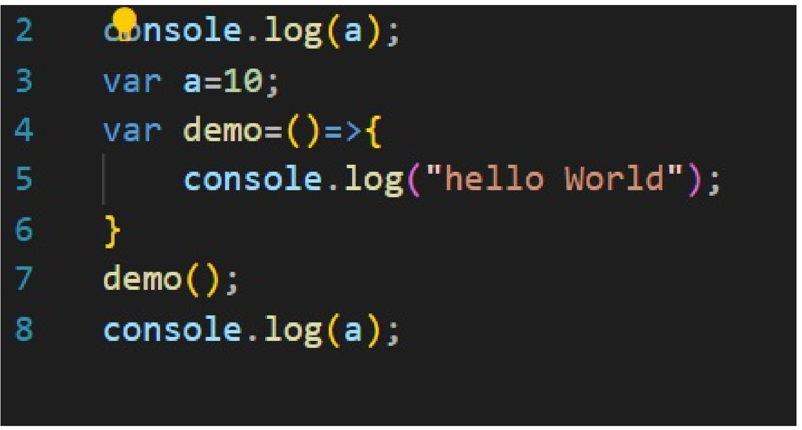


index1.js > ...

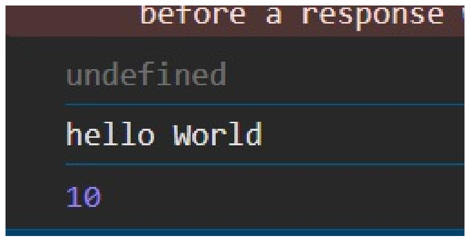
```
demo();
console.log(a);
var a=10;
var demo=()=>{
    console.log("hello World");
}
demo();
console.log(a);
```

Function as a variable

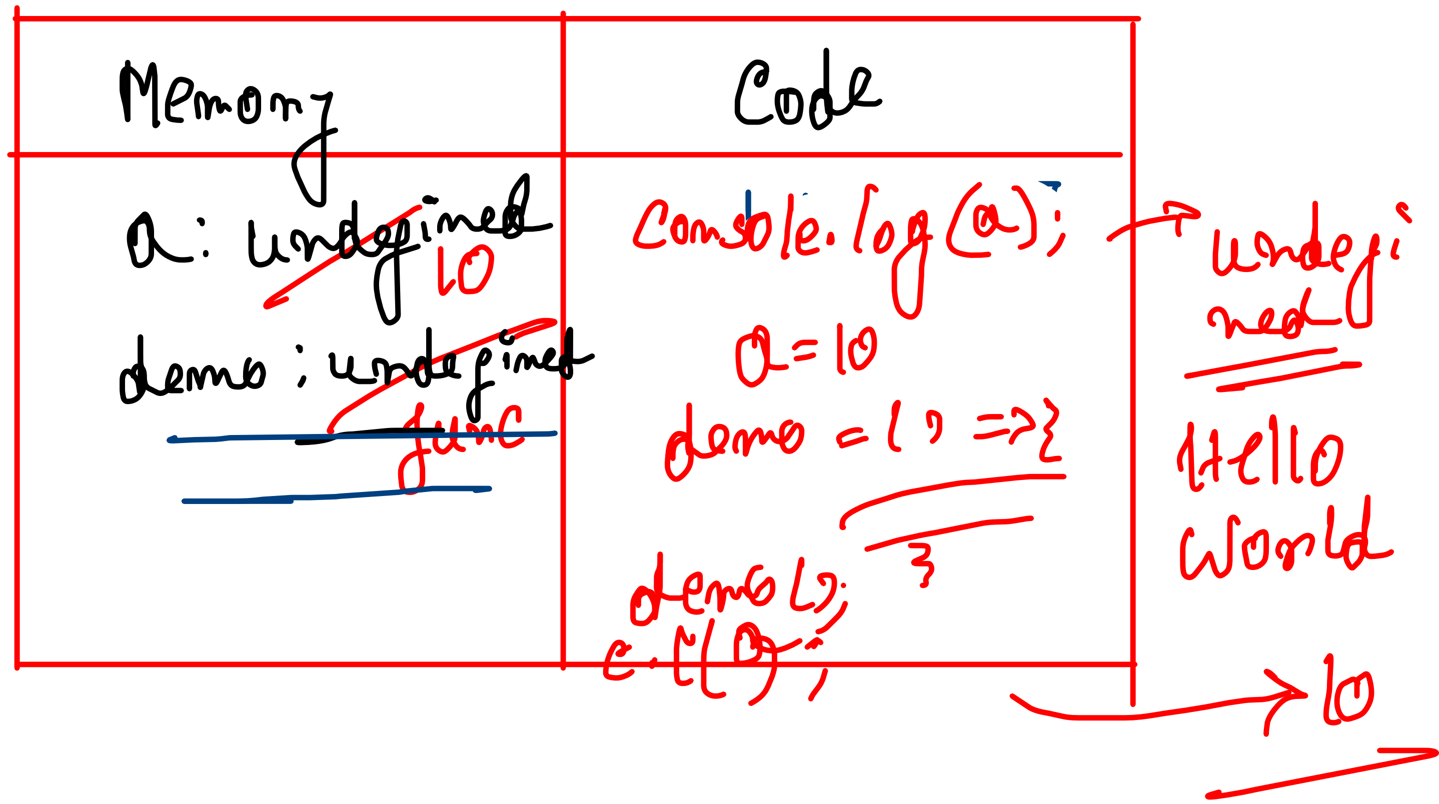
Memory	Code
a: undefined	demo()
demo: undefined	

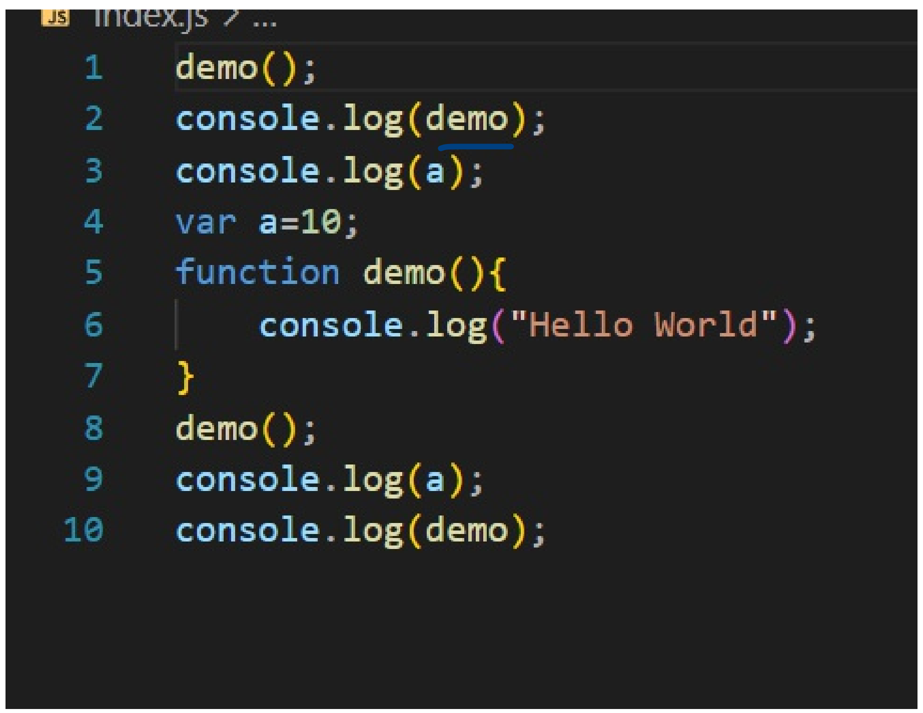


```
console.log(a);
var a=10;
var demo=()=>{
    console.log("hello World");
}
demo();
console.log(a);
```

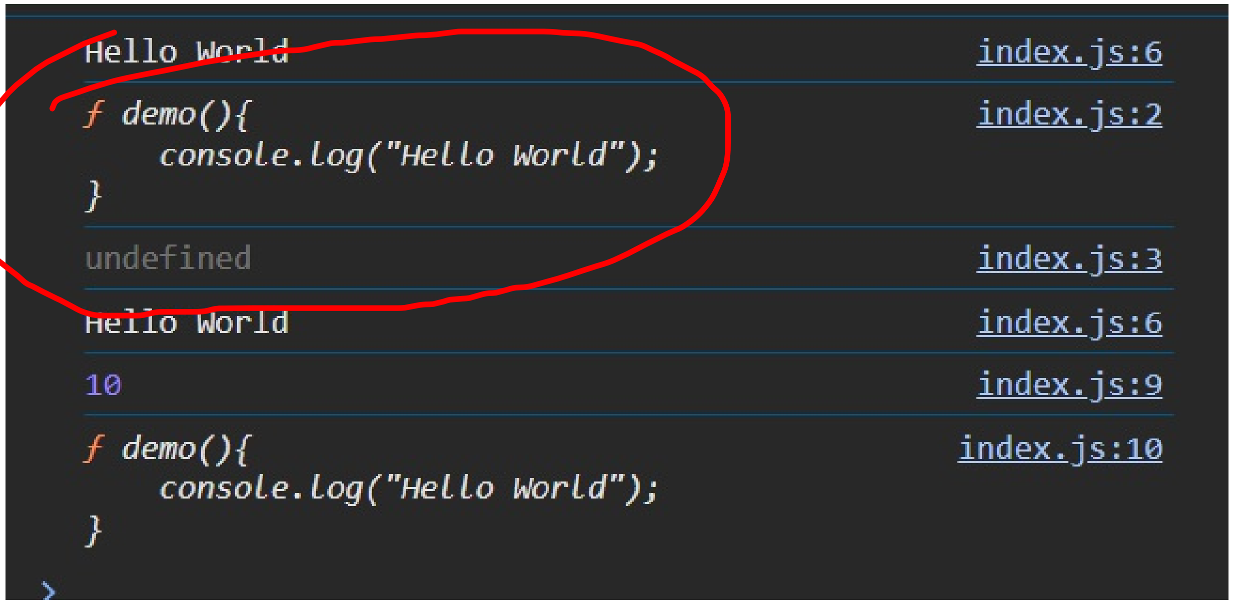


```
before a response
undefined
hello World
10
```

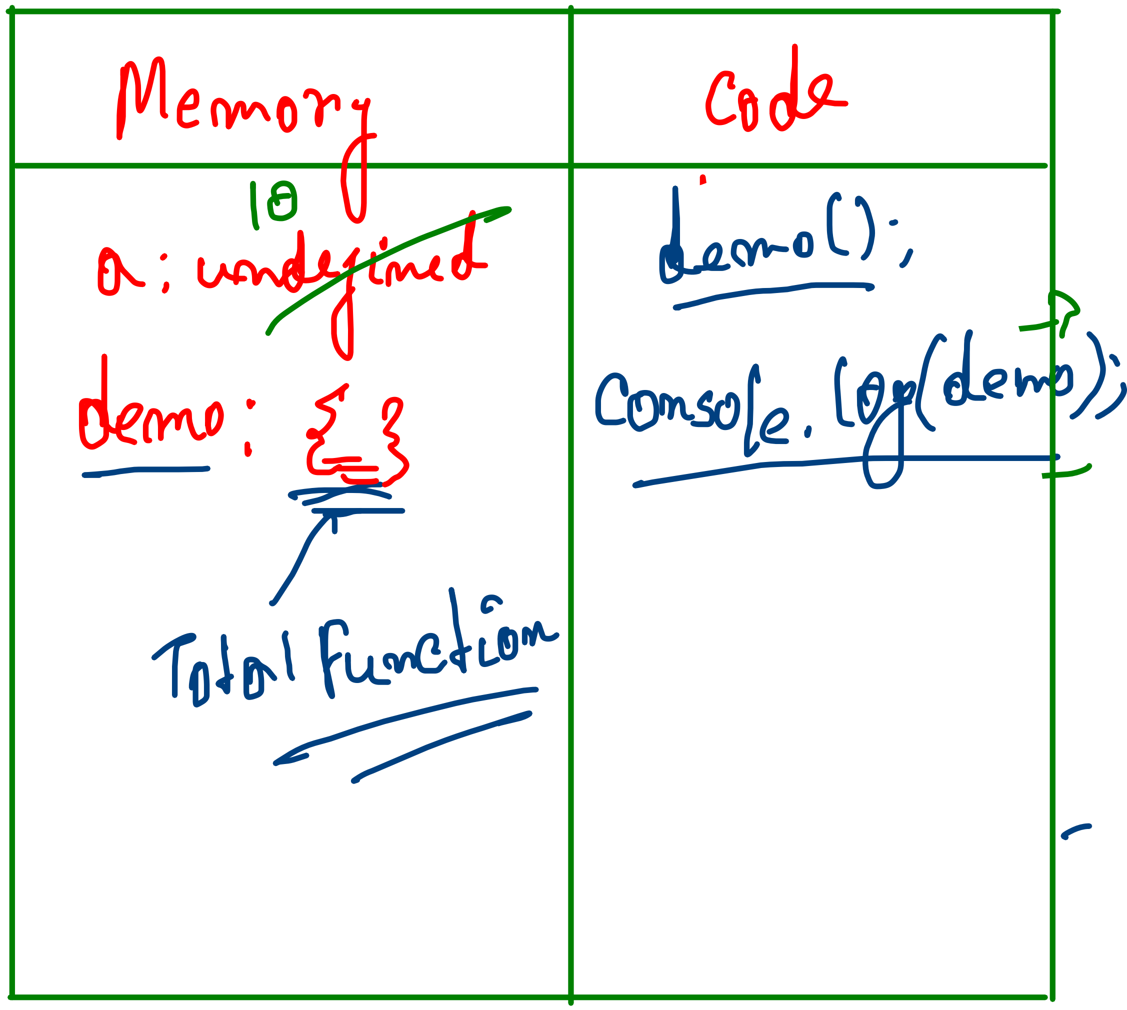




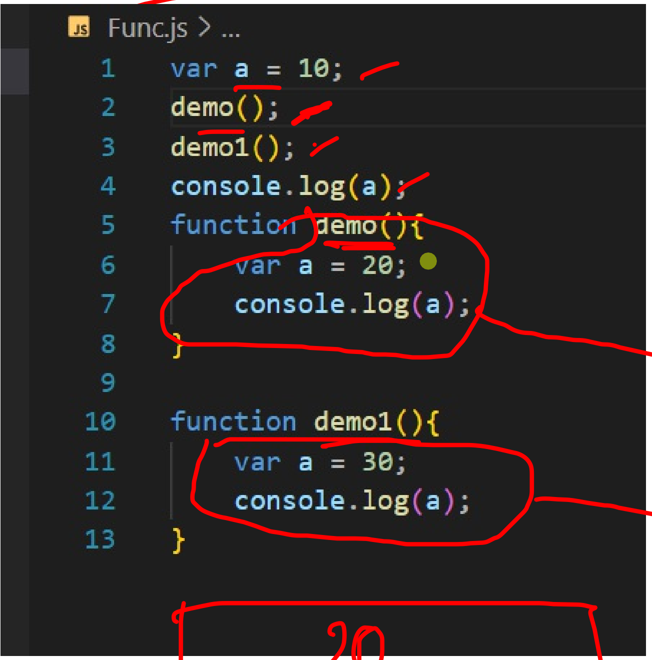
```
demo();
console.log(demo);
console.log(a);
var a=10;
function demo(){
    console.log("Hello World");
}
demo();
console.log(a);
console.log(demo);
```



```
Hello World                                        index.js:6
ƒ demo(){                                          index.js:2
    console.log("Hello World");
}
undefined                                          index.js:3
Hello World                                        index.js:6
10                                                 index.js:9
ƒ demo(){                                          index.js:10
    console.log("Hello World");
}
```



Function

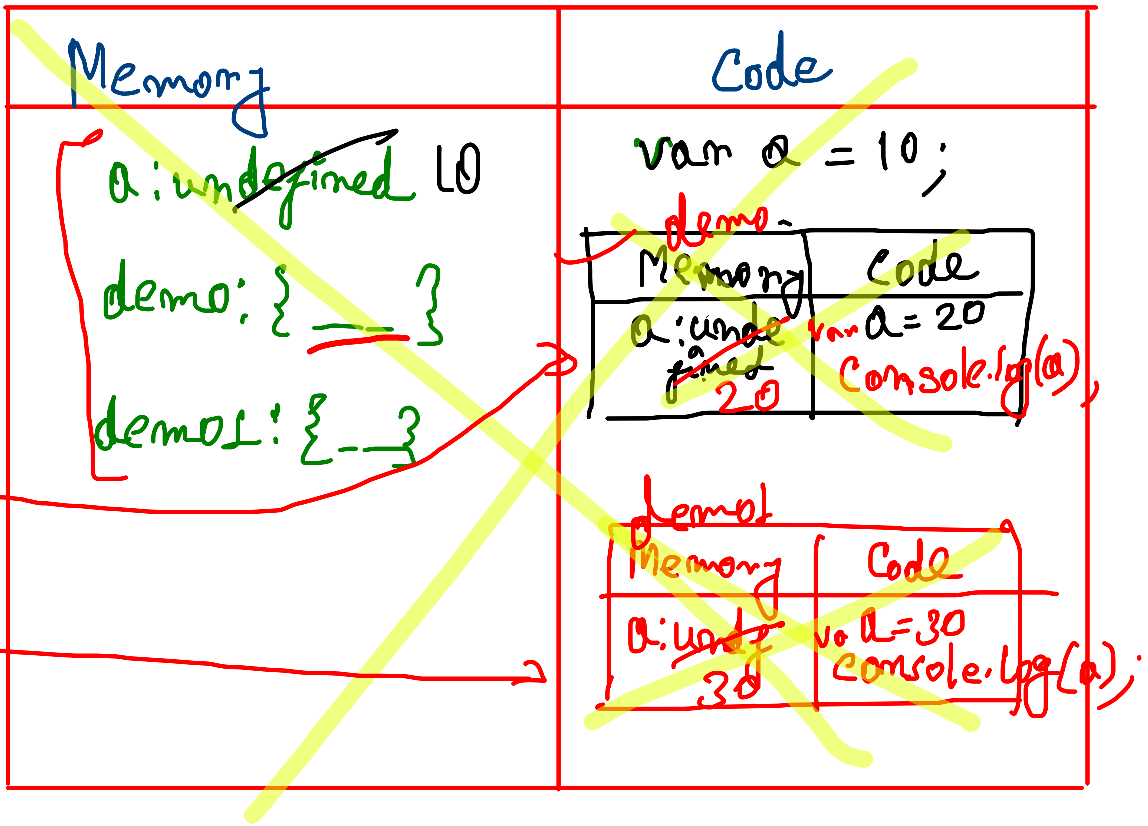


Func.js > ...

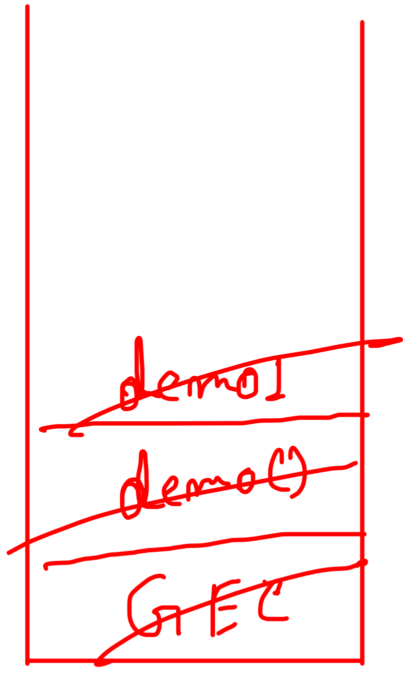
```javascript
var a = 10;
demo();
demo1();
console.log(a);
function demo(){
    var a = 20;
    console.log(a);
}

function demo1(){
    var a = 30;
    console.log(a);
}
```

20
30
10



Memory
Code
a : ~~undefined~~ 10
demo : { ___ }
demo1 : { ___ }
var a = 10 ;
demo
Memory
Code
a : ~~undefined~~ 20
var a = 20
console.log(a);
demo1
Memory
Code
a : ~~undef~~ 30
var a = 30
console.log(a);



~~demo1~~
~~demo()~~
~~GEC~~