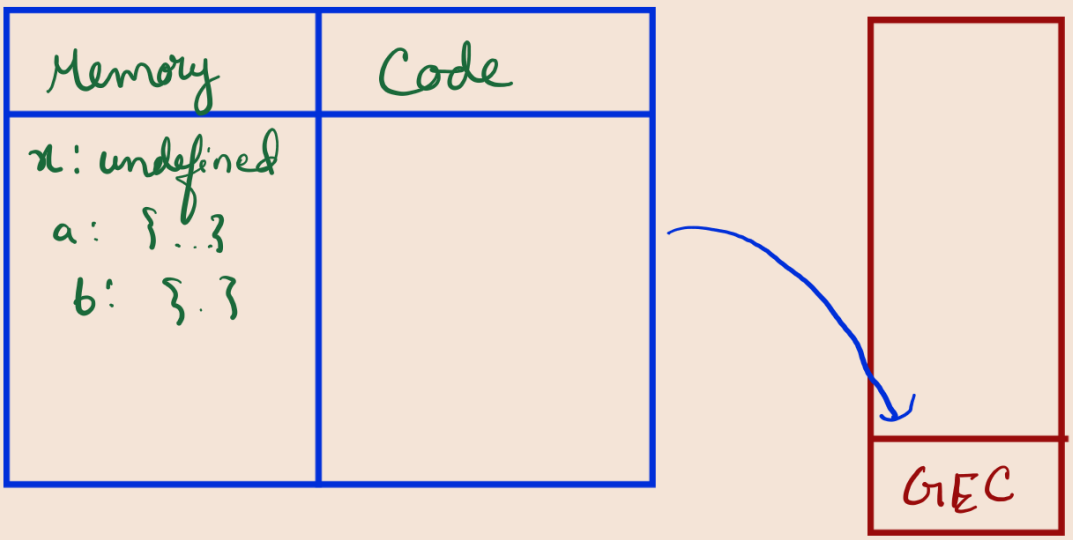
```
var x = 1;
a();
b();
function a(){
    var x = 10;
    console.log(x);
}

function b(){
    var x = 100;
    console.log(x);
}
```

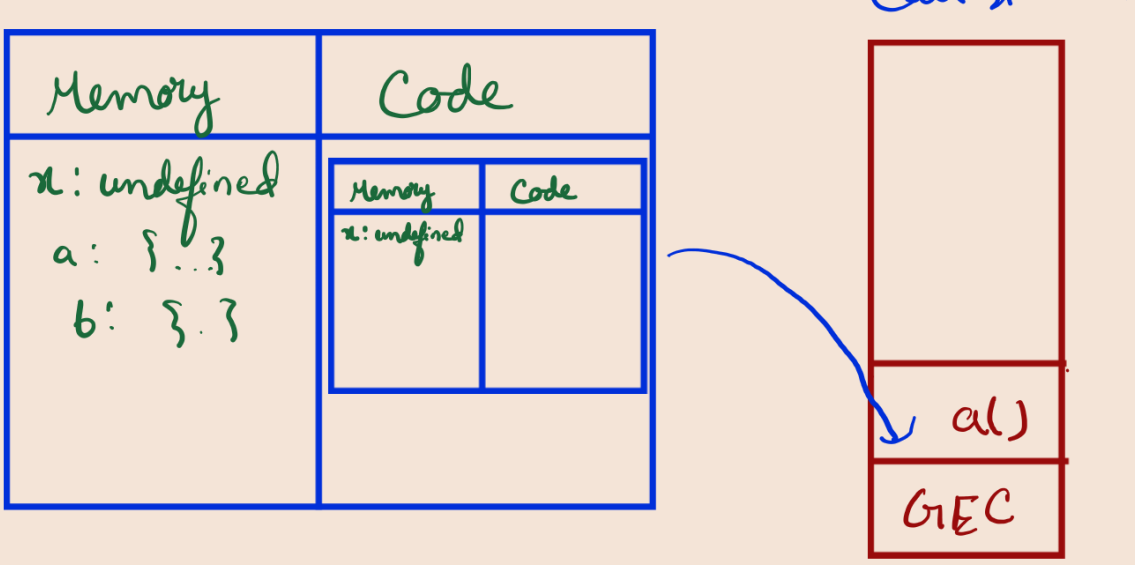
1) First phase is memory creation



2) 2nd phase now code is executed.

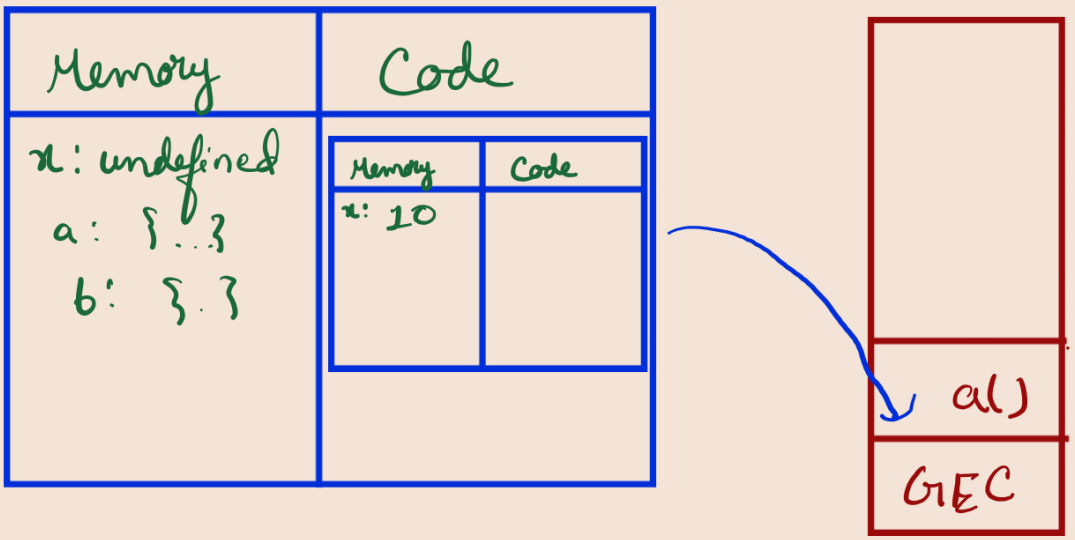
① line → It assigns x = 1

② line → function a is invocked, execution context is created → pushed in call stack.



The same steps will repeat inside execution context a()
- memory allocation of x as undefined.
- then it will turn the code and assign x = 10

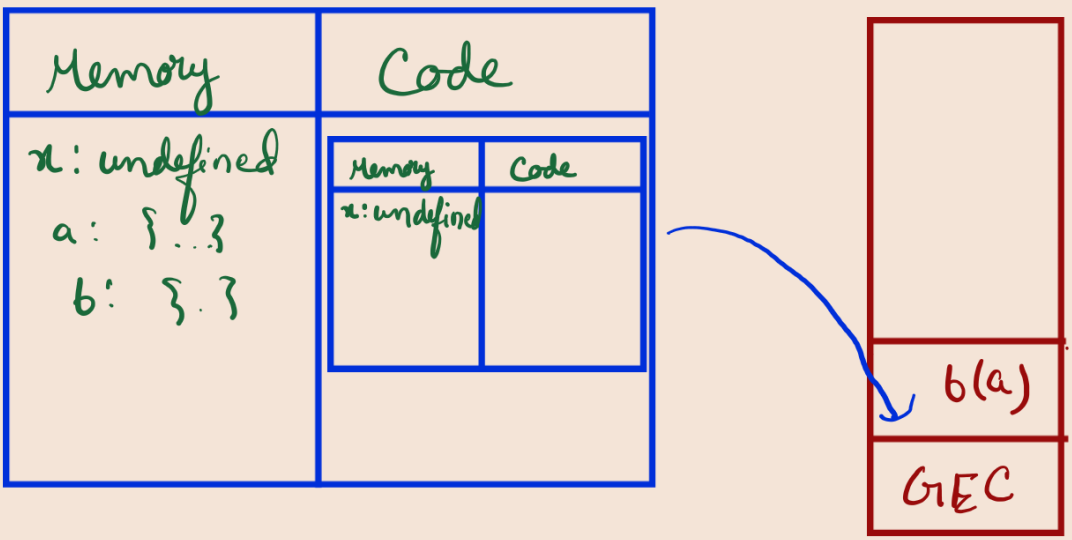
Call Stack.



- then it encounters next line and tries to find value of x in local memory space
- then prints 10 in console
- Execution is over, a() is popped out of call stack control goes back to GEC.

③ line → function b is invocked, execution context is created → pushed in call stack.

Call Stack.



Same steps repeats and 100 is printed on console.

⑤ line -> 4th line is executed. It will look for value of x in local space. Over here value of x=1 printed in console

When there's nothing more to execute, whole execution context is deleted, GEC is popped out from Call Stack.