* Function Invocation and Variable Environment :—

e.g. :—

```
1   var x = 1;
2   a();
3   b();
4   console.log(x);
5
6   function a(){
7     var x = 10;
8     console.log(x);
9   }
10
11  function b(){
12    var x = 100;
13    console.log(x);
14  }
```

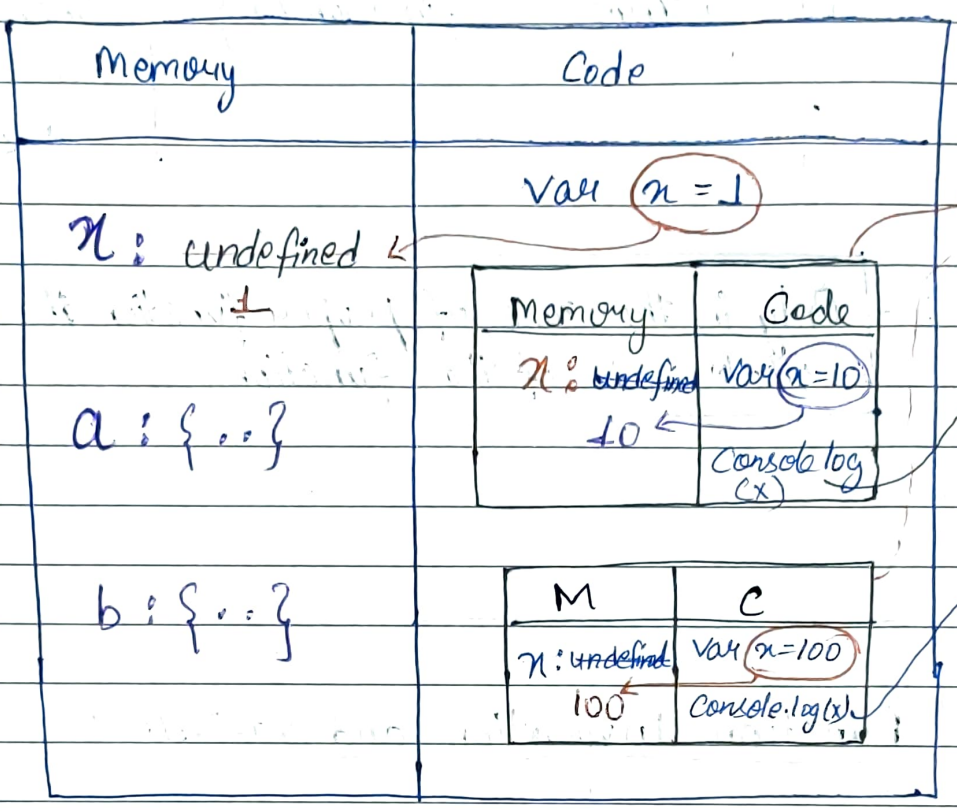
Output :—

10
100
1

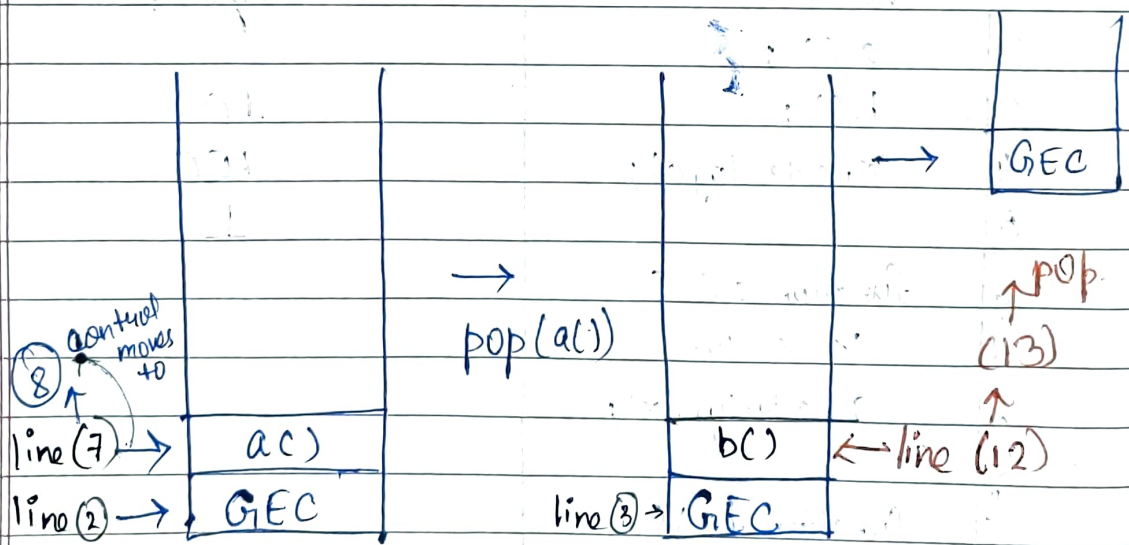
Function invocation in JavaScript means "calling" or "running" a function. When you invoke a function, you tell JavaScript to execute the code inside that function.

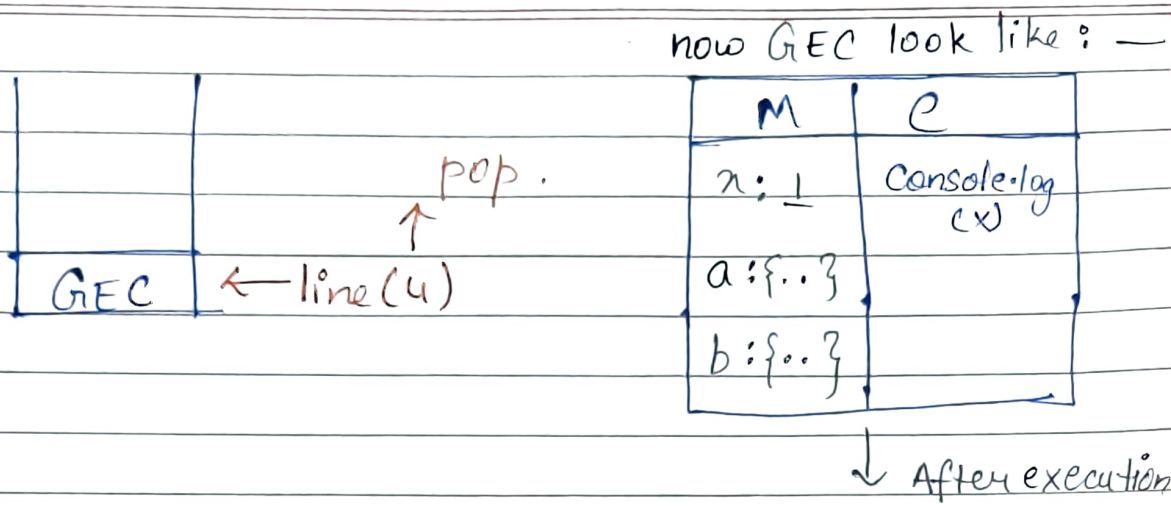
Defining a Function: function a() {
var x=10
consple.log(x); }
defines a function named 'a'.
This function will print 10 when called.
Similarly with function b(). => It will print 100 when called.

↳ Global Execution is created. (GEC)



Call stack :—





* After execution — the GEC is deleted & poped from stack.

A variable environment in JavaScript is like a storage box where your program keeps track of the variables (names and values) it uses while running.

Think of it like this:
Storage Box: When you create a variable (like let age = 25;), it's stored in the variable environment, just like putting something in a box.

Different Boxes for Different Places: Each time you run a function, a new "box" is created just for that function's variables. After the function finishes, this box is no longer needed and is thrown away.