* Scope:-

Scope means where we can access a specific variable & function in our code.

Scope in JavaScript is directly related to Lexical Environment

Example:-

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

→ Can we access b inside a?

Here JS will try to find out if 'b' exsits in the local memory space of a() or not.

→ & it won't be there. So what it will print?

o/p :— 10

→ It means somehow inside the f^n a(). The b is able to acess the b which is outside the f^n.

Example ② :— Here we are putting another f^n c() inside a() & trying to access b.

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

→ o/p :– 10

It means even inside the f^n which is inside another f^n which is inside the Global scope, we can access b ?

Example ③ :— We are checking here is vice versa true? Can we still acess b inside c()?

```
function a(){
 var b = 10;
  c();
 function c(){
 Console.log(b);
 }
} a();
```

→ o/p :– 10

Example ④ :— Here we are checking can we access b outside the f^n.

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

→ o/p: → b is not defined.

→ Here comes the scope into the picture.

→ Scope means where we can acess a specific variable & function. in our code.

→ There are 2 aspects to it.

1st) What is the scope of the variable 'b'?
(i.e. Where we can access this variable b).

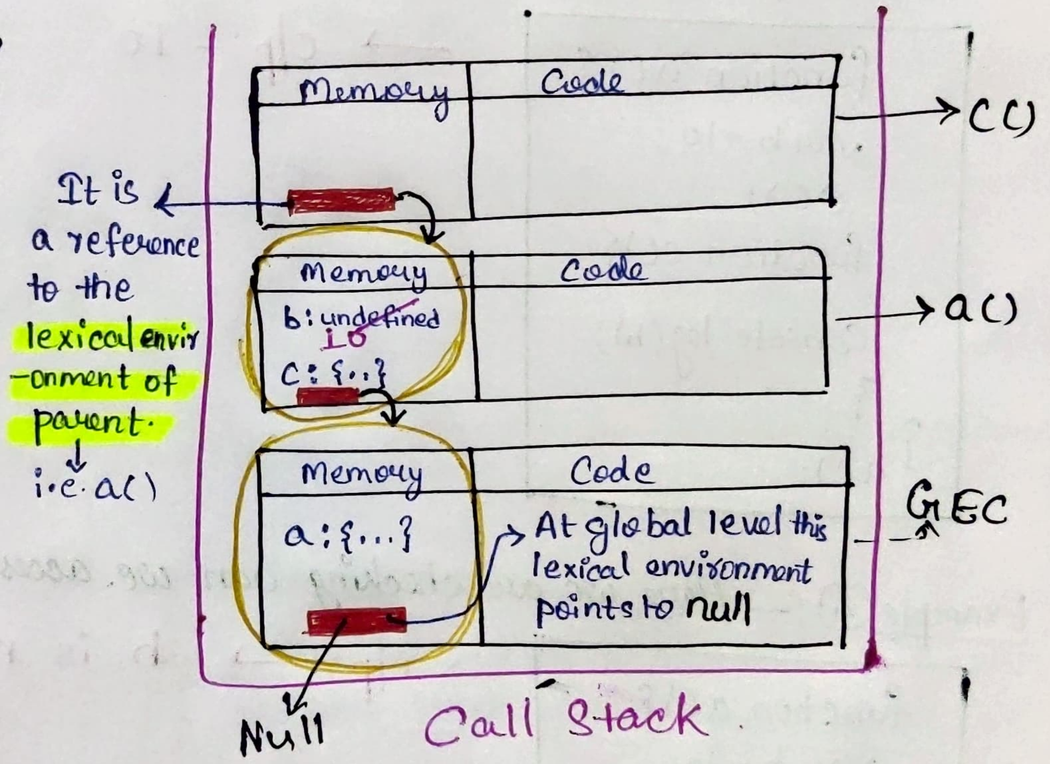
2nd) Is 'b' inside the scope of f^n c?
(i.e. Can I access this b inside c.)

→ These are one & the same thing. Just asking in different ways.

Q. What is Lexical Environment ?

→ We will going to learn what is lexical Enviroment with the help of visual representation.

```
e.g. function a(){
      var b=10;
      c();
      function c(){
      }
     }
     a();
     console.log(b);
```


It is a reference to the lexical environment of parent.
i.e. a()
Memory
Code
→ c()
Memory
b: ~~undefined~~ 10
c: {..}
Code
→ a()
Memory
a: {...}
Code
At global level this lexical environment points to null
GEC
Null
Call Stack

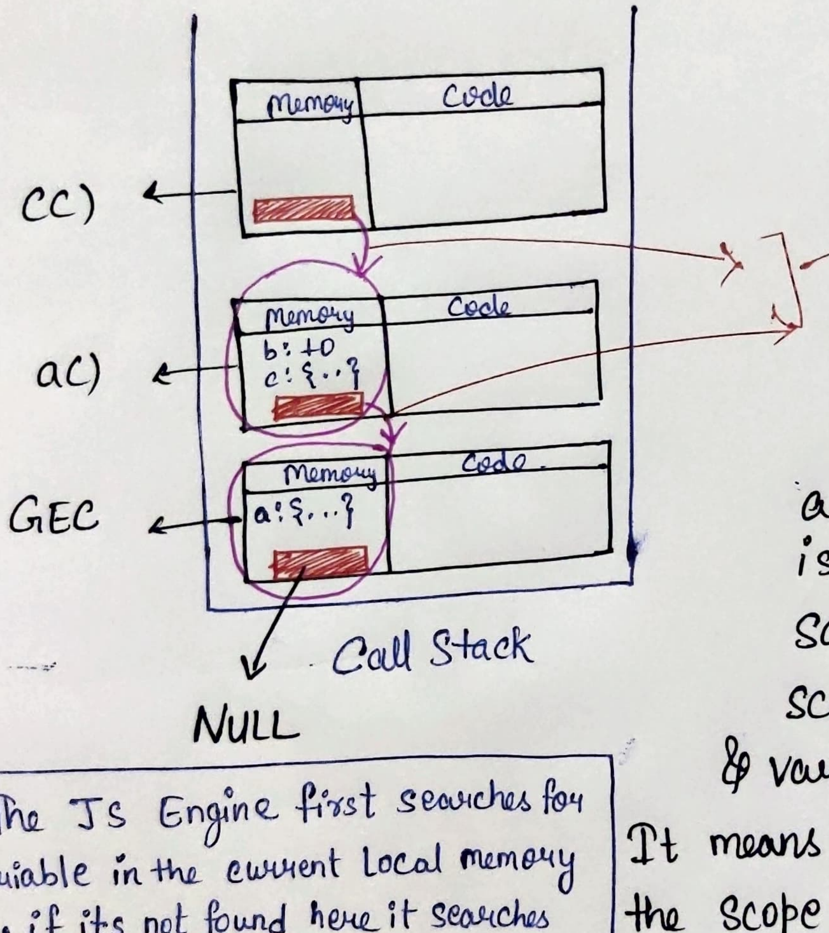
↳ Whenever an Execution Context is created, a lexical environment is created.

↳ Lexical environment is the local memory along with the lexical environment of its parent. { Lexical $\xrightarrow{\text{means}}$ In a hierarchy or in Sequence }

e.g. Above program.
→ We can say c() is lexically sitting inside a().

i.e. $\boxed{\text{Lexical Enviroment = local memory + Lexical Enviroment of parent.}}$

Que: What is Scope Chain?
→ Scope chain is nothing but the way of finding i.e. the chain of all the lexical environment & the parent references.



These whole Chain of lexical environments is known as SCOPE CHAIN. & it defines whethe a variable or ~~fil~~ functio is present inside the scope or no. If the scope chain is exhausted & variable is not found It means variable is not inside the scope chain.

→ The JS Engine first searches for a variable in the current Local memory Space, if its not found here it searches for the variable in the lexical environment of its parent, until the Variable is found in some lexical environment or the lexical enviromentbecomes NULL.