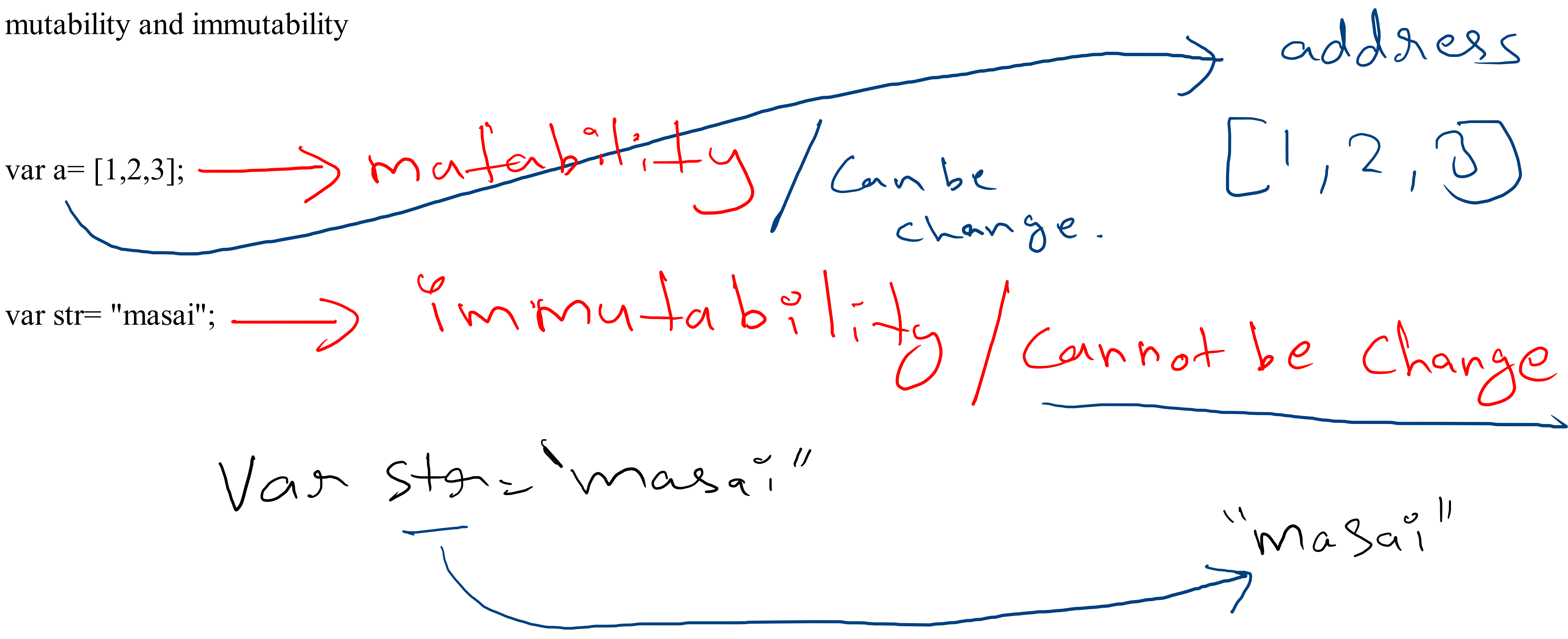
var a; ---> declaring.

var a= 10 ----> assigning

a= 20----> re-assigning

mutability and immutability



var a= [1,2,3]; → mutability / Can be change.

address

[1,2,3]

var str= "masai"; → immutability / Cannot be change

Var str='masai"

"masai"

scope

1. block scope.
2. global scope.
3. local or function scope.

{

}

var is global scope

let , const --> ES6

let, const are block scope.

Var a = 10

Var a = 20

```
→ console.log(a)
  var a;
```

hoisting---> Js will have access to all the variable declaration and function declaration before the code execute.

Exeation context

memory phase

Code execution

```
var a
name = {
    console.log("1")
}
```

```
name()
```

```
function name(){
    console.log("1")
}
```

object ---> key and value pair

keys are always unique.

let obj = {

a : ~~1~~ 2,

name : "mahesh"

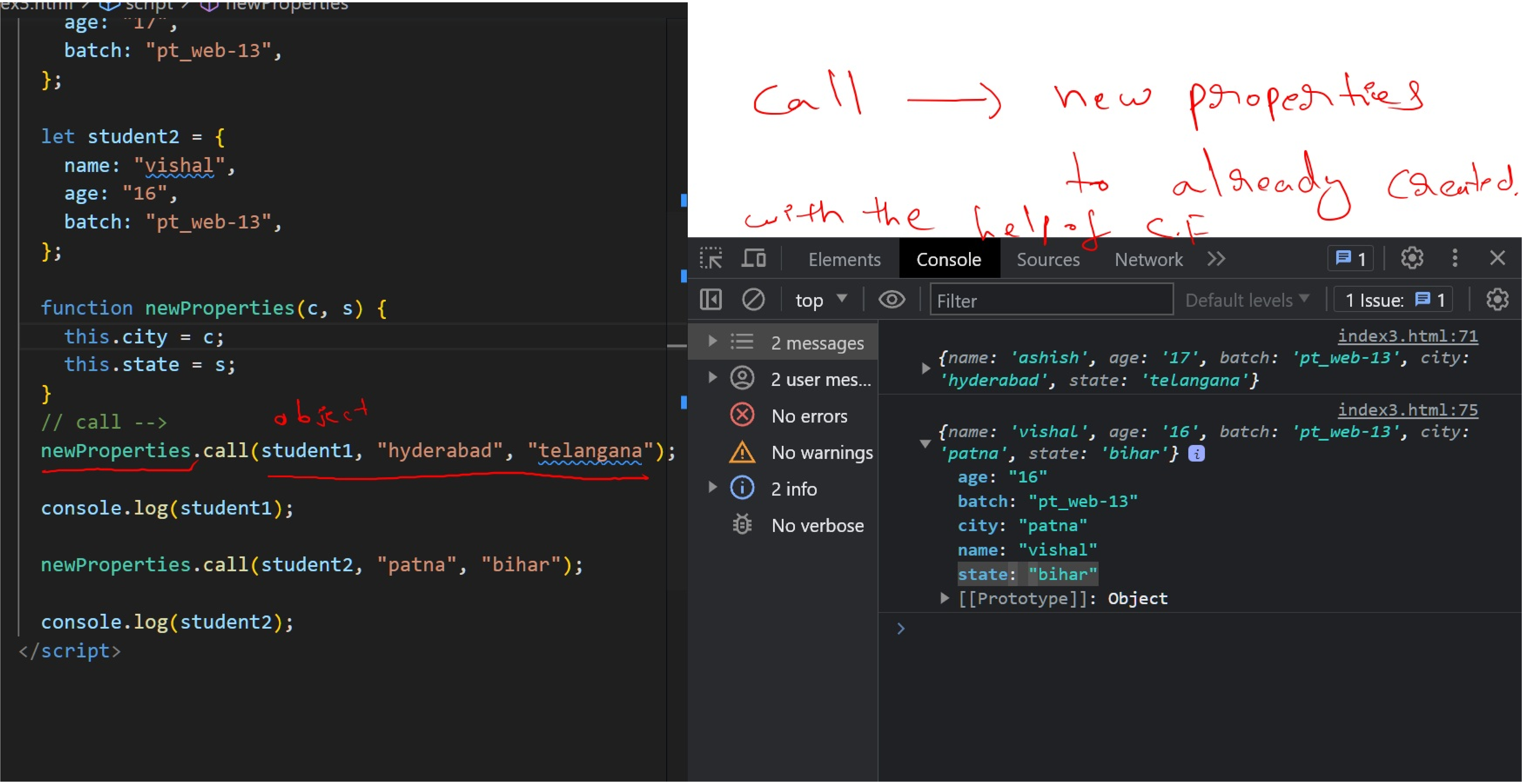
}

obj["a"] = 2 | obj.name = "mahesh"

Constructor → Custom object

```
Let stud1 = {
        name :
        age :
        city :
        state :
}
```

Call ⟶ new properties
to already created
with the help of C.F



```
    age: "17",
    batch: "pt_web-13",
  };

  let student2 = {
    name: "vishal",
    age: "16",
    batch: "pt_web-13",
  };

  function newProperties(c, s) {
    this.city = c;
    this.state = s;
  }
  // call -->
  newProperties.call(student1, "hyderabad", "telangana");

  console.log(student1);

  newProperties.call(student2, "patna", "bihar");

  console.log(student2);
</script>
```

object

```
index3.html:71
▶ {name: 'ashish', age: '17', batch: 'pt_web-13', city:
  'hyderabad', state: 'telangana'}
index3.html:75
▼ {name: 'vishal', age: '16', batch: 'pt_web-13', city:
  'patna', state: 'bihar'}
    age: "16"
    batch: "pt_web-13"
    city: "patna"
    name: "vishal"
    state: "bihar"
  ▶ [[Prototype]]: Object
>
```

Constructor function → Custom object

Call, apply, bird

We can add new properties to the already present object with the help of call, apply, bird.

call ⟶ all new properties will be seperated by ,

apply ⟶ ["masai", "course"]

bind ⟶ all new properties will be seperated by ,

↳ we need to call the function inorder to add properties.

object

adding function inside the object---> method.

constructor function ---> custom object.

method---> call, apply, bind.