Call, Bind & Apply

```
function test1() {
    console.log(this);
}
```

```
let obj = {
    test2() {
        console.log(this);
    }
}
```

* test1(); → window
* obj.test2(); → obj

this is happening because of implicit binding of function

function random() {

the this key word inside function should be forcefully changed

}

let obj3 = {

}

* functionName.call(object)

↓

it should be the function which you want to bind with any other obj.

↳ the object with which you want to bind your fn.

```
function intro() {
    console.log(this.age);
}
```

```
intro();
```

* window.age

→ undefined

```
let obj = { age: 24 }
```

```
intro.call(obj);
```

↳ 24

* call → it's a method of function, which explicitly binds function to any given obj.

functionName.call(objName)

```
function intro (firstName) {
    console.log(`Hi my name is ${firstName}
        - ${this.lastName}`)
}
```

* intro.call(obj);
* intro.call(obj, firstName);

function test(arg1, arg2, arg3){

}

---

* test.call(obj, arg1, arg2, arg3);
