



Aulinha da Emi

POO em Javascript



O que é uma classe?

criando

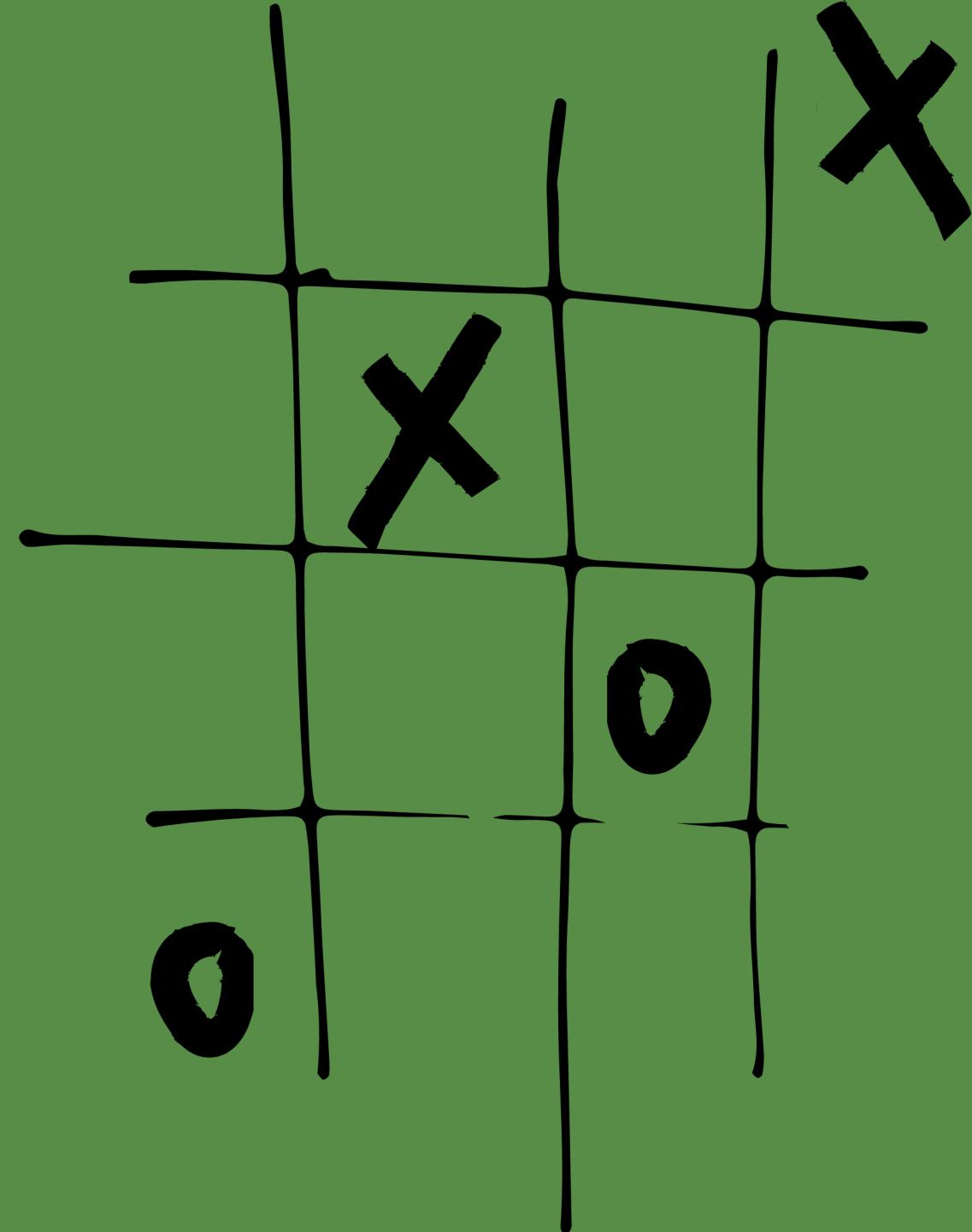
classes

```
class Pessoa:  
    def __init__(self, nome, sexo, cpf):  
        self.nome = nome  
        self.sexo = sexo  
        self.cpf = cpf  
  
    def falar(self):  
        print(f'{self.nome} esta falando...')  
  
    def comida(self, comida):  
        print(f'{self.nome} esta comendo {comida}')  
  
if __name__ == "__main__":  
    pessoa1 = Pessoa("Joao", "M", "1234566778")  
    print(pessoa1.nome)  
    print(pessoa1.sexo)  
    print(pessoa1.cpf)  
    pessoa1.comida('arroz')
```

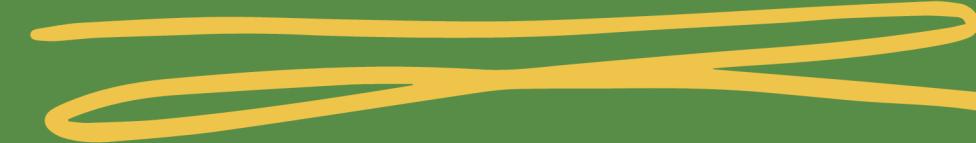
Criando uma class em Python

Criando uma class em JS

```
class Pessoa {  
    constructor(nome, sexo, cpf){  
        this.nome = nome;  
        this.sexo = sexo;  
        this.cpf = cpf;  
    }  
  
    falar(){  
        console.log(`${this.nome} está falando.`)  
    }  
  
    comer(comida){  
        console.log(`${this.nome} está comendo ${comida}`)  
    }  
}  
  
const Pessoa1 = new Pessoa('Marcelo', 'M', '121230230')  
console.log(Pessoa1)  
console.log(Pessoa1.nome)  
Pessoa1.falar()  
Pessoa1.comer("arroz")
```



que tal **fixar**
o conteúdo?



Exercício 1:

Faça uma **classe carro**, com: marca, modelo e ano.
Essa classe contem o método buzinar.

Exercício 2:

Faça uma **classe livro**, com: titulo, autor e
anoPublicacao. Essa classe contem o método
resumo, no qual deve imprimir (ex): "Os dois
morrem no final' é um livro de Adam Silveira
publicado em 2017"

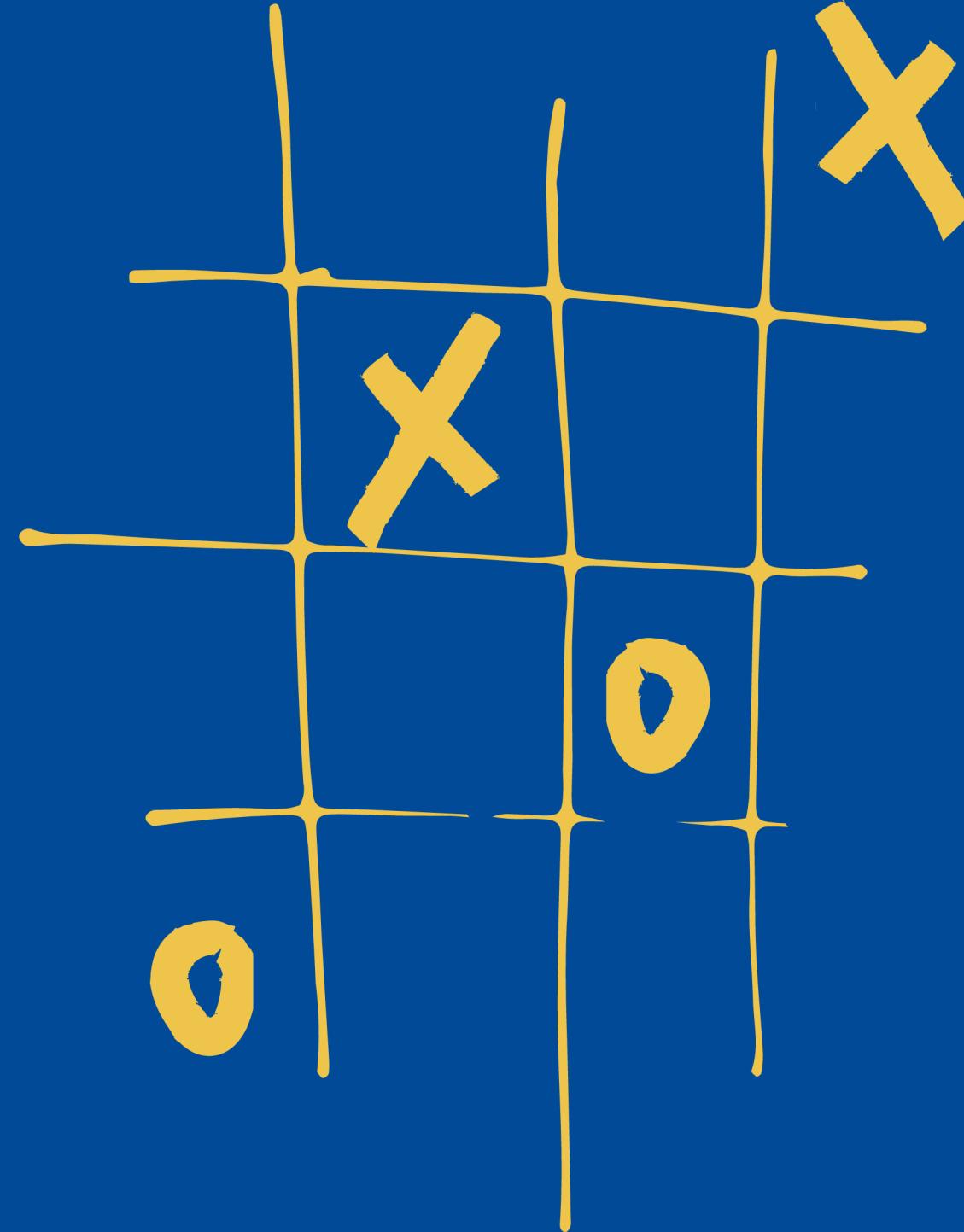
vamos criar getters/setters?

```
1 class Pessoa:  
2     def __init__(self, nome, idade):  
3         self.nome = nome  
4         self.__idade = idade  
5  
6     def get_idade(self):  
7         return self.__idade  
8  
9     def set_idade(self, idade_nova):  
10        self.__idade = idade_nova  
11  
12 if __name__ == "__main__":  
13     pessoa1 = Pessoa('João', 20)  
14     print(pessoa1.get_idade())  
15     pessoa1.set_idade(34)  
16     print(pessoa1.get_idade())  
17
```

Getter e Setters em Python

```
class Pessoa{  
    #idade; // atributo privado  
  
    constructor(nome, idade){  
        this.nome = nome  
        this.#idade = idade  
    }  
  
    get idade(){  
        return this.#idade  
    }  
  
    set idade(idadeNova){  
        this.#idade = idadeNova  
    }  
  
}  
  
pessoal = new Pessoa('João', 20)  
console.log(pessoal.idade)  
pessoal.idade = 34  
console.log(pessoal.idade)
```

Getter e Setters em JS



que tal **fixar**
o conteúdo?



Exercício 3:

Pegue a **classe livro** do exercício anterior e atribua **get** e **set** nele.

Exercício 4:

Faça uma **classe conta bancária**, onde o saldo é um **atributo privado**. Deve ter um método com condicional onde o saldo **não** pode ser negativo.

E como funciona
Herança?



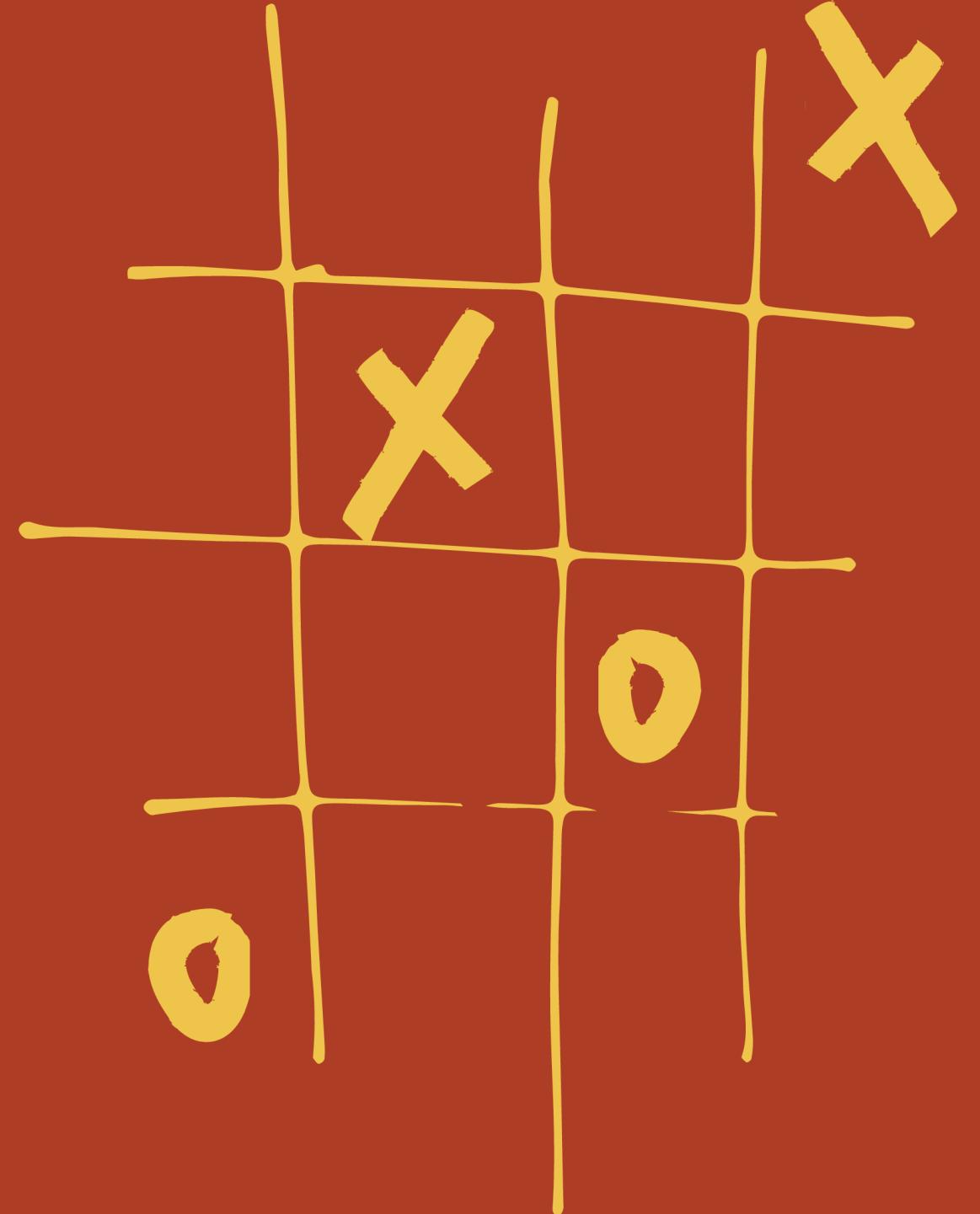
```
1 class DispositivoEletronico{  
2     constructor(nome){  
3         this.nome = nome;  
4         this.ligado = false;  
5     }  
6  
7     ligar(){  
8         if(!this.ligado){  
9             this.ligado = true;  
10            console.log(` ${this.nome} ligado.`)  
11        }  
12        else{  
13            console.log(` ${this.nome} já esta ligado.`)  
14        }  
15    }  
16  
17    desligar(){  
18        if(!this.ligado){  
19            console.log(` ${this.nome} já desligado.`)  
20        }  
21        else{  
22            this.ligado = false  
23            console.log(` ${this.nome} acabou de desligar.`)  
24        }  
25    }  
26  
27    falaTchau(){  
28        console.log("Tchau")  
29    }  
30  
31 }  
32
```

```
// extends é o que chama do que smartphone terá herança
class Smartphone extends DispositivoEletronico{
    constructor(nome, cor, modelo){
        super(nome); // chamou o construtor da classe DispositivoEletronico
        this.cor = cor;
        this.modelo = modelo;
    }

    ligar(){
        console.log('Olha, podemos também sobrepor métodos do super! Isso se chama polimorfismo')
    }

    falaOi(){ // método sómente do Smartphone
        console.log('Oi')
    }
}

const s1 = new Smartphone('Samsung', 'Preto', 'Galaxy S10')
console.log(s1)
```



que tal **fixar**
o conteúdo?



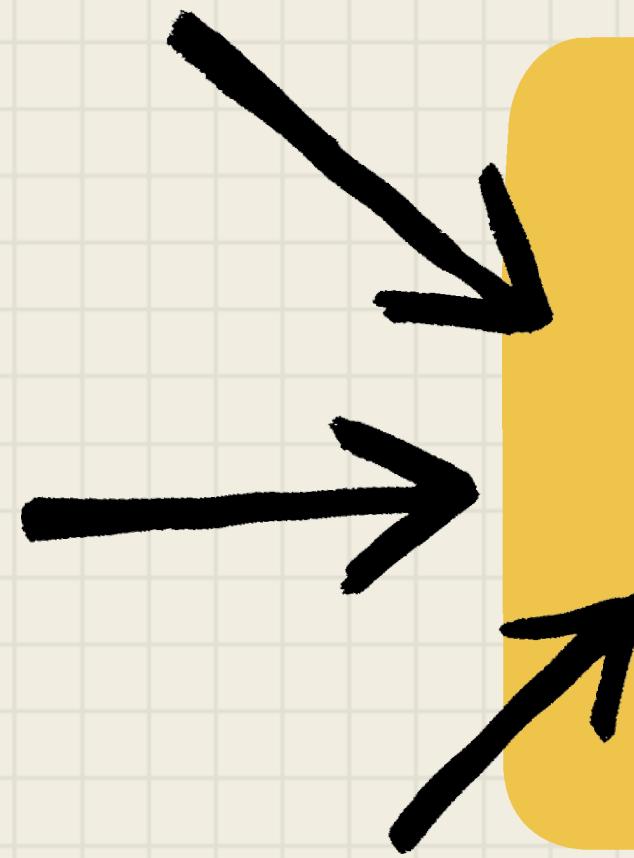
Exercício 5:

Crie uma **classe base** chamada **Funcionario** com os atributos nome e salario. Crie uma **subclasse chamada Gerente** que **herda de Funcionario** e adiciona um atributo adicional chamado departamento. Crie um método na subclasse Gerente para exibir todas as informações (nome, salário e departamento).

Vamos de...

KAHoot !!!

CÓDIGO



vamos para o....
Exercício final?

SuperClasse: Crie uma classe base chamada **Veículo**, com atributos: marca, modelo e ano e um método de `mostrarInformacoes()`.

SubClasses: Teremos a **classe Carro** com método de `abrirJanelas()` e teremos a **classe Moto** com o método `empinar()`.

Encapsulamento: O atributo ano deve ser encapsulado e acessível somente através de **getters** e **setters**.

Polimorfismo: Utilize o método mostrarInfo() para diferentes tipos de veículos.

Muito obrigada
pela atenção!

Tenham um bom dia <3