	INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO PIAUÍ **Curso: ADS** **Disciplina: Programação Orientada a** **Objetos Professor: Ely**

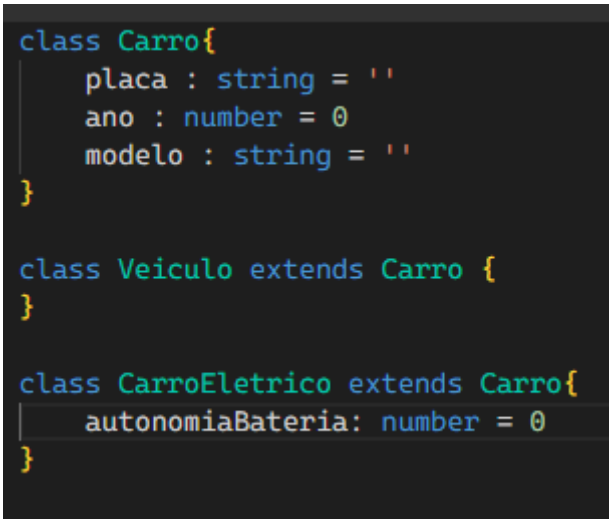
Exercício 07

1. As classes **Carro, Veiculo** e **CarroEletrico** são bem semelhantes. Reescreva as classes usando herança para que os atributos duplicados não sejam mais necessários.

```
class Veiculo {
    placa: String;
    ano: number;
}
```

```
class Carro {
    placa: String;
    ano: number;
    modelo: String;
}
```

```
class CarroEletrico {
    placa: String;
    ano: number;
    modelo: String;
    autonomiaBateria:
number; }
```



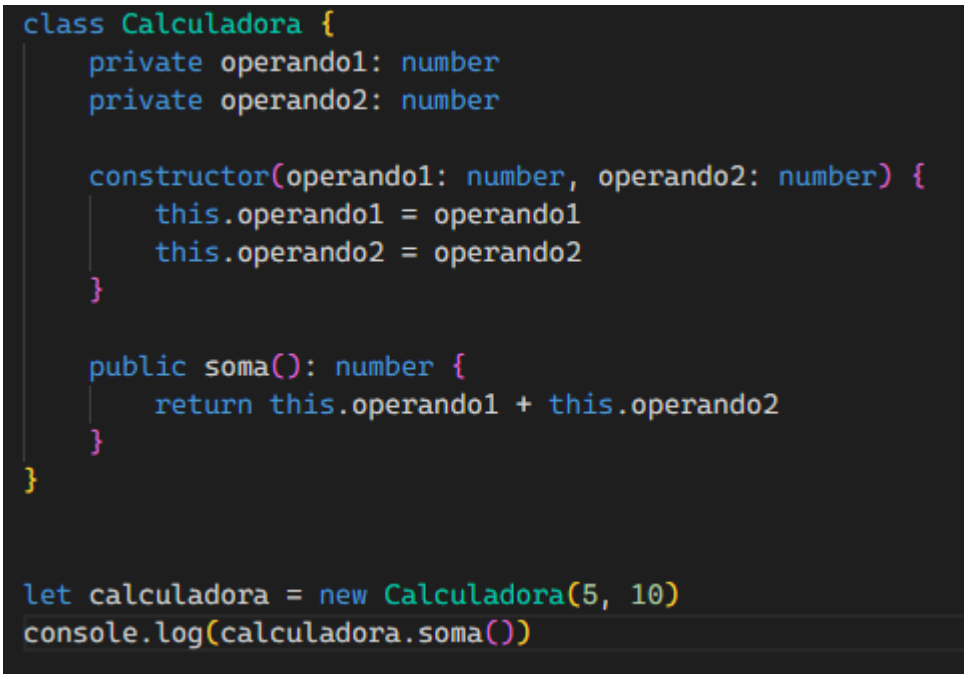
```
class Carro{
    placa : string = ''
    ano : number = 0
    modelo : string = ''
}

class Veiculo extends Carro {
}

class CarroEletrico extends Carro{
    autonomiaBateria: number = 0
}
```

Código:
https://www.typescriptlang.org/play?#code/MYGwhgzhAEDCYCcEHsDeAoaXoAdzDGgC5oIAXBASwDsBzaAXmgHJnNsxrljpqBXALYAjAKYJG0AAzssA5ABMRIbiXJU6E1ugC+6dKEgwAaiMrA+y6CIAeZEdXkx4Sbhl37wUOIhQBRECIUZtw2dg5OPmgy0GB8ZMhcApRgAEJgdlRgJPzCYhLS2kA

2. Crie uma classe Calculadora com:
    a. Dois tributos privados chamados representando dois operandos;
    b. Crie um construtor que inicializa os atributos;
    c. Crie um método que retorna a soma dos dois atributos;
    d. Teste a classe.



```
class Calculadora {
    private operando1: number
    private operando2: number

    constructor(operando1: number, operando2: number) {
        this.operando1 = operando1
        this.operando2 = operando2
    }

    public soma(): number {
        return this.operando1 + this.operando2
    }
}


let calculadora = new Calculadora(5, 10)
console.log(calculadora.soma())
```

Código:https://www.typescriptlang.org/play?#code/MYGwhgzhAEDCYmAV3AEwPYCczQN4ChojoAHTASwDcwAXAU2nRLuwDsMBGALmlaQFsARi0LEyVWgyYsw7dACYefISNFFg6VhBqYkwGlgAU0tpyUDhmADSNmpheZWYAlHjXEiNABbkIAOhNZTmgAXlsZOQ53D29fALsghVDw+3l3AF98dxlkQRByYGgIdH4wQ2dHSzcPD0w6GiRMVmhY-0DI6ABqFp82hLk0j0zMrJB66GAEZDQsHDDWOgB3OCmUMAxsQwBWGw4ABmd8DS10Mb8QdABzQ0nENY2wP2LS8tcgA

3. Crie uma classe chamada Calculadora Científica que herda da classe Calculadora do exercício passado e:
    a. Implemente um método chamado exponenciar que retorne o primeiro operando elevado ao segundo;
    b. Teste a classe;
    c. Foi necessária alguma modificação em Calculadora para o acesso aos atributos?

```
class Calculadora {
    protected operando1: number // permite que classes filhas acessem os atributos
    protected operando2: number

    constructor(operando1: number, operando2: number) {
        this.operando1 = operando1
        this.operando2 = operando2
    }

    public soma(): number {
        return this.operando1 + this.operando2
    }
}

class CalculadoraCientifica extends Calculadora{
    constructor(operando1:number, operando2 : number){
        super(operando1, operando2)
    }

    public exponenciar() : number{
        return Math.pow(this.operando1 , this.operando2)
    }
}

let calcientifica = new CalculadoraCientifica(2, 4)
```

a modificação necessária foi a alteração de “private’ para “protected” no momento da atribuição dos operandos, permitindo que a classe filha conseguisse utilizá-los

código:https://www.typescriptlang.org/play?#code/MYGwhgzhAEDCYmAV3AEwPYCczQN4ChojoAHTdAFwFNhrVp0SrsA7DARgC5oWkBbAEbNoAehGlmfAJbVoARyRVooSBCowAZlJAALSNDDB1avgxhgKmKQKQV0EQsTKUadBk1YYATN16Dm+I5EwOgsEJZItFgAFIzMYGzoXDz8QpgANO7xiT4p-pgAlHhBxEQUOllQAHRxnknQALxZdewlpeWVNR4J3o3NPeheJQC+gaUkSAIgUsDQEOh8YNEFvqnCBKWlmFQUSJgs0B3VtQPs0ADUhxXH3Tkj+KP4KlBwCMhoWGCwUlQsFFJaYA4KgAD2obBg8EQKDAGGwG2IITCESimFitw4nD8aUyJxy0FW+QKCM2cyQHnR2Q4uIxgwK91KJQmUxm0FBJFCv2AUjAaKKhLSJM2212+2gAFkLDoqhyAO7RI5dKn1TKKvHeemlUaPEA7ZRvH5-AEzHBNFhUWWvaEfbDfX7-QFLLyZAAsBSAA
```

4. Considerando a implementação da aplicação bancária, implemente: a. Implemente na classe Banco o método renderJuros(numero: string):  number, onde:
    i. É passado por parâmetro o número de uma poupança e feita uma consulta para ver se a conta existe. Note que a consulta não se altera sendo Conta ou Poupança;
    ii. Caso a poupança seja encontrada, teste se realmente se trata de uma poupança com o operador instanceof, desconsidere a operação caso contrário;
    iii. Caso seja, faça um cast e invoque o método renderJuros da própria instância encontrada;
    iv. Teste o método da classe Banco passando tanto um número de poupança como de conta passados inseridos anteriormente;
```

v. Altere a aplicação anteriormente sugerida para ter a opção de menu "Render Juros".

b. Adicione a aplicação para também permitir o cadastro da ContaImposto feita em sala de aula;

c. Incremente a implementação da aplicação para recuperar de um arquivo texto para o array contas salvas em um arquivo contas.txt com um formato semelhante ao abaixo:

111-1; 40; C
222-2; 10.65, CP; 0.5
333-3; 2.00; CI; 0.38
444-4; 140; CP; 0.5

Onde os campos separados por ponto-e-vírgula são o número, o saldo, o tipo da conta e, no caso de conta imposto e conta poupança, a taxa de desconto e taxa de juros.

Pesquise uma biblioteca de leitura e escrita de arquivos e deixe essa e a próxima opção disponíveis para o usuário escolher

d. Implemente também uma funcionalidade de gravar no mesmo arquivo o conteúdo do array de contas

5. Suponha um sistema de controle de estoque de produtos e implemente:

a. Duas classes: Produto e ProdutoPerecivel;

b. A classe Produto tem atributos privados representando identificador, descrição, quantidade de produtos em estoque e valor unitário;

c. ProdutoPerecivel tem as mesmas características de Produto, porém possui a mais um atributo representando a data da validade (https://www.javatpoint.com/typescript-date-object). Use herança;

d. Produto possui dois métodos para repor e dar baixa. A e ambos somam e subtraem respectivamente uma quatidade passada por parâmetro do atributo quantidade;

e. Um produto perecível possui um método que diz se um produto está válido ou não comparando sua data de validade com a data atual;

f. Use sobrescrita, ou seja, reescreva os métodos de inserir, repor e dar baixa para que não seja possível executar a ação caso o produto não esteja na validade;

g. Crie uma classe chamada Estoque que possui um atributo privado representando um array de produtos (Produto ou ProdutoPerecivel); h. Implemente métodos para inserir, consultar pelo atributo id, excluir, repor e dar baixa nos produtos na classe estoque;

i. Crie validações para não deixar serem incluídos produtos com mesmo id ou mesmo nome;

j. Os métodos repor e dar baixa na classe estoque chamam os métodos da

classe produto finalmente alterar a quantidade;

k. Os vários métodos da classe devem levar em conta se o produto existe, para isso, use o método consultar. Caso precise, crie métodos de consulta auxiliares;

l. Implemente um método que liste todos os produtos perecíveis vencidos.

Código:

https://www.typescriptlang.org/play?#code/MYGwhgzhAEAKBOB7AJgVwC6OgbwFDQOgAd4BLANzHQFNpTkAuaCdMgOwHMBufQkiqrWTUIwMsDCImLdt14F+lGtACOqMG3T0wwpm1QBbAEbV4PQsTJLalEIngBVNqXRgyU6PuOme86MEQ2GVRgTHgACnppVlJOABpoYVFxSWjZBLUNLWQdaj1DE3gE23snFzdSDy9CgEocPwt0AAtSCAA6emgAXjpkcwtCZta2pLFSCSwe0ZTEfoHoIfbMzW1hbtV1FZzhOYHFtpLHZ1d3dcOyk8rdgF9cP3hqIntw5ezc-O94Orx5gn3X1a0ADUPQB22oNzuFhy8AAQmBSAAPMAvTZvXSeAqmb4NQYtJZowHQAC0oMJ4Mht1woEgMAQKAwiFgpmowAo1BA0GoiJobGQdKQaEw9QsikEiSoYAAamAQICmAARQS+CwBIKsEJhSKMZgxeKJERjCZpWIcDLk96Yz7FWWlY4VKpYooS1wyuXgxWCHG-ZioIimbUJabjSTmrKAm12I7ldw1XaNfEjSVuoITZOywGQvwiV0Z5CIcLe35qljQJqIABWtB6bGoAHdoEqaIX44QHuhUPA2AtEzlc+7ctAAHxlysQvxUiwPJ4RMGW6rYkW-UgAM2g4X2OeleYLNSLPoIED9pja0+ec+EcdxBFuFknhBh8KRKIveSttSX81X683LG37t3fcDyPf14CTOEEWRVFw3BK8fVvQhbipGkoGgABRFhEDUWgfj4KxxRIBlMAgJgAEF4HgMAAE8AB56SFLAAB84EFRlmQeNlyA5EcegAbQAXRVQhYggUxSAiQiGKYejGWgZiZMwdjWXZEAgIsAB6dToBTHIAHOAGOsCINwwE8Qy6DYUBUHEywiMQGAAgMaAsAMEQDCwTpEFQTxEFc69oBXex1xAah0GIZy132STGQgNTlzXcIiA6ZBui6HposwZK5OYpLgwmVL0tYzK8skOKD2gdtOzYVt5gQ2r-Kior7LaIhUAgJpEqauDEKhQgS1QEATm1E1ODKirQqqnthgy5qV1iZBOrs7oRxmrK0p6ehupvXqCG5KzxOG3VZDGkswvm7l1kauz2jmvkAEk+W5RaGOW2yGLWtLei2gZv0iR7EWgABCT7iQARjGhNpqa9oICIOVgGoP7hERBJwZq6A6oxnbxpnQ6ZFNMMtnnJ0TsCUsZsuxN+sGtxtW+ixfpmiG8Ls09HnPC1L3RzH7wIR9IJRKIjoJjYYOJz5SfVN7ZJ6fZqaGzb0cZrrPwPVb+efaCia5-yeexuU-3gBT7KUzjqFaKVqEs+h7MLVWpwmrspvaGabtIQaA380UmteimRNcSzqEQNdjdNlToAAMgjoHVq3HTd3876kKAA