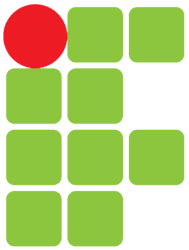

INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
PIAUÍ

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO PIAUÍ
TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS
MÓDULO II - PROGRAMAÇÃO ORIENTADA A OBJETOS
PROF.: Ely Miranda
ALUNO: Lucas Gomes de Oliveira (20191ADS0185)

Exercício 05

1. Crie uma classe `Calculadora` que tenha:

a. Dois atributos privados (`operando1` e `operando2`) do tipo number;

b. Dois métodos públicos, cada um representando uma operação básica;

c. Um construtor onde são passados os operandos e que esses inicializam os atributos;

Teste a classe calculadora e seus métodos. Tente acessar os atributos diretamente e verifique o que acontece.

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

    constructor(x: number, y: number) {
```

```
        this.operando1 = x;
        this.operando2 = y;
    }

    somar(): number {
        return this.operando1 + this.operando2;
    }

    multiplicar(): number {
        return this.operando1 * this.operando2;
    }
}

let operacao = new Calcuradora(5, 7);

console.log(`Resultado da soma => ${operacao.somar()}`);
console.log(`Resultado da soma => ${operacao.multiplicar()}`);
```

Na tentativa de se localizar os atributos (variáveis) que pertencem ao objeto, não é possível realizar qualquer modificação ou acesso com o intuito de atualizar os valores.

2. Crie uma classe Hora que tenha:

a. Três atributos privados e definidos no construtor chamados hora, minutos e segundos;

b. Métodos públicos para ler hora, minuto e segundo de forma individual;

c. Um método público para retorne a hora no formato “hh:mm:ss”.

```
class Hora {
    private hora;
    private minuto;
```

```
    private segundo;

    constructor(hora: number, minuto: number, segundo: number) {
        this.hora = hora;
        this.minuto = minuto;
        this.segundo = segundo;
    }

    lerHora(): string {
        return this.hora.toString() + ':';
    }

    lerMinuto(): string {
        return this.minuto.toString();
    }

    lerSegundo(): string {
        return ':' + this.segundo.toString();
    }
}

let horario = new Hora(24, 5, 12);
console.log(`Horario => ${horario.lerHora() + horario.lerMinuto() +
horario.lerSegundo()}`)
```

3. Altere as implementações da classe `Banco` das aulas anteriores para que:

- **a. O array de contas seja privado;**
- **b. O método consulta por índice seja privado;**
- **c. Os demais métodos sejam públicos.**

```
class Conta {
    numero: String;
    saldo: number;

        constructor(numero: String, saldoInicial: number) {
        this.numero = numero;
        this.saldo = saldoInicial;
    }

    sacar(valor: number): void {
        if (this.saldo >= valor) {
            this.saldo = this.saldo - valor;
        }
    }

    depositar(valor: number): void {
        this.saldo = this.saldo + valor;
    }

    transferir(contaDestino: Conta, valor: number): void {
        this.sacar(valor);
        contaDestino.depositar(valor);
    }

}

class Banco {
    contas: Conta[] = [];
```

```
    inserir(conta: Conta): void {
            let contaConsultada = this.consultar(conta.numero);

            if (contaConsultada == null) {
            this.contas.push(conta);
            }
    }

    consultar(numero: String): Conta {
        let contaConsultada: Conta;
        for (let conta of this.contas) {
            if (conta.numero == numero) {
                contaConsultada = conta;
                break;
            }
        }
        return contaConsultada;
    }

    private consultarPorIndice(numero: String): number {
        let indice: number = -1;
        for (let i: number = 0; i < this.contas.length; i++) {
            if (this.contas[i].numero == numero) {
                indice = i;
                break;
            }
        }
        return indice;
    }
```

```
    alterar(conta: Conta): void {
        let indice: number = this.consultarPorIndice(conta.numero);
        if (indice != -1) {
            this.contas[indice] = conta;
        }
    }

    excluir(numero: string): void {
        let indice: number = this.consultarPorIndice(numero);

        if (indice != -1) {
            for (let i: number = indice; i < this.contas.length; i++)
{
                this.contas[i] = this.contas[i+1];
            }

            this.contas.pop();
        }
    }

    depositar(numero: String, valor: number): void {
        let contaConsultada = this.consultar(numero);

        if (contaConsultada != null) {
            contaConsultada.depositar(valor);
        }
    }
```

```
    sacar(numero: String, valor: number): void {
        let contaConsultada = this.consultar(numero);

        if (contaConsultada != null) {
            contaConsultada.sacar(valor);
        }
    }

    transferir(numeroCredito: string, numeroDebito: string, valor:
number){
        let contaCredito: Conta = this.consultar(numeroCredito);
        let contaDebito: Conta = this.consultar(numeroDebito);

        if (contaCredito != null && contaDebito != null) {
            contaDebito.transferir(contaCredito, valor);
        }
    }

    calcularQuantidadeContas(): number {
        return this.contas.length;
    }

    calcularTotalSaldos(): number {
        let totalSaldo: number = 0;
        for (let conta of this.contas) {
            totalSaldo += conta.saldo;
        }

        return totalSaldo;
```

```
    }

    calcularMediaSaldos() {
        return
this.calcularTotalSaldos()/this.calcularQuantidadeContas();
    }
}


let objetoConta: Conta = new Conta("1", 200)
let objetoBanco: Banco = new Banco();

objetoBanco.inserir(objetoConta);

// teste
// objetoBanco.inserir(new Conta("1", 150));
objetoBanco.sacar("1", 20);
console.log(objetoConta.saldo);

objetoBanco.inserir(new Conta("2", 350));
objetoBanco.inserir(new Conta("3", 1090));
objetoBanco.inserir(new Conta("4", 100));

console.log(objetoBanco.consultar("2").saldo);

objetoBanco.transferir("4", "1", 10);

console.log(objetoBanco.calcularQuantidadeContas());
console.log(objetoBanco.calcularTotalSaldos());
```

```
console.log(objetoBanco.calcularMediaSaldos());
```

4. Altere também a sua classe Conta dos exercícios anteriores para:

a. Ter atributos privados e métodos "get" para leitura;

b. Verifique se sua implementação da classe Banco e os testes precisarão ser adaptados para ter métodos de leitura e escrita, visto que os atributos que agora são privados.

```
class Conta {
    private _numero: String;
    private _saldo: number;

    constructor(numero: String, saldoInicial: number) {
        this._numero = numero;
        this._saldo = saldoInicial;
    }

    sacar(valor: number): void {
        if (this._saldo >= valor) {
            this._saldo = this._saldo - valor;
        }
    }

    depositar(valor: number): void {
        this._saldo = this._saldo + valor;
    }

    transferir(contaDestino: Conta, valor: number): void {
        this.sacar(valor);
        contaDestino.depositar(valor);
    }
```

```
    get getNumero(): String {
        return this._numero;
    }

    get getSaldo(): number {
        return this._saldo;
    }
}

class Banco {
    private _contas: Conta[] = [];

    inserir(conta: Conta): any {
            let contaConsultada = this.consultar(conta.getNumero);

            if (contaConsultada == null) {
                this._contas.push(conta);
            } else {
                console.log("Impossivel adicionar uma mesma conta");
                console.log(`Operacao => ${conta.getNumero}`);
            }
    }

    consultar(numero: String): Conta {
        let contaConsultada: Conta;
        for (let conta of this._contas) {
            if (conta.getNumero == numero) {
                contaConsultada = conta;
                break;
            }
```

```
        }
        return contaConsultada;
    }

    private consultarPorIndice(numero: String): number {
        let indice: number = -1;
        for (let i: number = 0; i < this._contas.length; i++) {
            if (this._contas[i].getNumero == numero) {
                indice = i;
                break;
            }
        }
        return indice;
    }

    alterar(conta: Conta): void {
        let indice: number = this.consultarPorIndice(conta.getNumero);
        if (indice != -1) {
            this._contas[indice] = conta;
        }
    }

    excluir(numero: string): void {
        let indice: number = this.consultarPorIndice(numero);

        if (indice != -1) {
            for (let i: number = indice; i < this._contas.length; i++) {
                this._contas[i] = this._contas[i+1];
            }

            this._contas.pop();
```

```
        }
    }

    depositar(numero: String, valor: number): void {
        let contaConsultada = this.consultar(numero);

        if (contaConsultada != null) {
            contaConsultada.depositar(valor);
        }
    }

    sacar(numero: String, valor: number): void {
        let contaConsultada = this.consultar(numero);

        if (contaConsultada != null) {
            contaConsultada.sacar(valor);
        }
    }

    transferir(numeroCredito: string, numeroDebito: string, valor:
number){
        let contaCredito: Conta = this.consultar(numeroCredito);
        let contaDebito: Conta = this.consultar(numeroDebito);

        if (contaCredito != null && contaDebito != null) {
            contaDebito.transferir(contaCredito, valor);
        }
    }

    calcularQuantidadeContas(): number {
        return this._contas.length;
```

```
    }

    calcularTotalSaldos(): number {
        let totalSaldo: number = 0;
        for (let conta of this._contas) {
            totalSaldo += conta.getSaldo;
        }

        return totalSaldo;
    }

    calcularMediaSaldos() {
        return this.calcularTotalSaldos()/this.calcularQuantidadeContas();
    }
}


let objetoConta: Conta = new Conta("1", 100)
let objetoBanco: Banco = new Banco();

objetoBanco.inserir(objetoConta);
console.log(`Saldo da primeira conta  => ${objetoConta.getSaldo}`)

// teste
//objetoBanco.inserir(new Conta("1", 150));
objetoBanco.sacar("1", 20);
console.log(objetoConta.getSaldo);

objetoBanco.inserir(new Conta("2", 350));
objetoBanco.inserir(new Conta("3", 1090));
objetoBanco.inserir(new Conta("4", 100));
```

```
console.log(objetoBanco.consultar("2").getSaldo);

objetoBanco.transferir("4", "1", 10);

console.log(objetoBanco.calcularQuantidadeContas());
console.log(objetoBanco.calcularTotalSaldos());
console.log(objetoBanco.calcularMediaSaldos());
```