//MODULES:

//Os módulos foram criados para que pudéssemos interagir entre arquivos e elementos javascript. Para isso usamos as palavras "export" e "import":

//Primeiro de tudo, é impossível trabalhar com módulos se a nossa pasta de arquivos não tiver um arquivo "package.json", pois precisamos explicitar para o ES que o vamos trabalhar com módulos, e isso só é possível se o nosso package.json tiver a chave "type" com o valor "module", dessa forma:

/\*      {

            "name": "javascript",

            "version": "1.0.0",

            "description": "",

            "main": "index.js",

            "dependencies": {

              "lodash": "^4.17.21"

            },

            "devDependencies": {},

            "scripts": {

              "test": "echo \"Error: no test specified\" && exit 1"

            },

            "author": "",

            "license": "ISC",

            "type": "module"

        }

 \*/

//Depois que tivermos explicado que vamos trabalhar com módulos precisamos de usar 2 palavras reservadas do javascript "export" para exportar um elemento que está num módulo, e "import" para capturar elementos que estão num módulo externo para o nosso módulo interno.

//Abra o arquivo "182-Module\_part2.js" para ver como fizemos para exportar a classe "Pessoa" para funcionar como classe mãe da classe "ContaCorrente" que temos no nosso arquivo...

import {Pessoa} from './182-Module\_part2.js' //Veja a sintáxe que usamos para importar somente a classe Pessoa.

                                                //Poderíamos importar uma classe inteira usando o sinal de "\*" mas isso

                                                //não é muito indicado tendo em vista que ficaria muito pesada a

                                                //execução do nosso arquivo...

class ContaCorrente extends Pessoa {

    constructor(nome, cpf, rg, conta, agencia, saldo){

        super(nome, cpf, rg)

        this.conta = conta

        this.agencia = agencia

        this.saldo = saldo

    }

    deposito(valor){

        this.saldo += valor

        console.log(`\nVocê depositou R$ ${valor.toFixed(2)}`)

        console.log(`Saldo atual R$ ${this.saldo.toFixed(2)}`)

    }

    saque(valor){

        if(this.saldo >= valor){

            this.saldo -= valor

            console.log(`\nSaldo restante R$ ${this.saldo.toFixed(2)}`)

        } else {

            console.log(`\nSeu saldo é insuficiente para sacar ao valor de R$ ${valor.toFixed(2)}`)

            console.log(`Saldo atual R$ ${this.saldo.toFixed(2)}`)

            console.log(`Se deseja sacar mesmo assim use o comando saqueCredito()`)

        }

    }

    saqueCredito(valor){

        this.saldo -= valor

        console.log(`\nSaldo restante R$ ${this.saldo.toFixed(2)}`)

    }

    verSaldo(){

        console.log(`\nSaldo: R$ ${this.saldo.toFixed(2)}`)

    }

}

//Instanciando

let client = new ContaCorrente('João', '124.235.457.8', '235.124.457.8', '00001', '25145-1', 1500 )

console.log(client)

/\*RESULTADO NO CONSOLE:

  ContaCorrente {

  nome: 'João',

  cpf: '124.235.457.8',

  rg: '235.124.457.8',

  conta: '00001',

  agencia: '25145-1',

  saldo: 1500

}

\*/

//Testando métodos:

//Métodos herdados de pessoa...

client.dadosPessoais()

/\*RESULTADO NO CONSOLE:

Nome: João

CPF: 124.235.457.8

RG: 235.124.457.8

 \*/

client.verSaldo()

/\*RESULTADO NO CONSOLE:

Saldo: R$ 1500.00

\*/

client.deposito(1000)

/\*RESULTADO NO CONSOLE:

Você depositou R$ 1000.00

Saldo atual R$ 2500.00

\*/

client.saque(500)

/\*RESULTADO NO CONSOLE:

Saldo restante R$ 2000.00

\*/

client.saque(2500)

/\*RESULTADO NO CONSOLE:

Seu saldo é insuficiente para sacar ao valor de R$ 2500.00

Saldo atual R$ 2000.00

Se deseja sacar mesmo assim use o comando saqueCredito()

\*/

client.saqueCredito(2500)

/\*RESULTADO NO CONSOLE:

Saldo restante R$ -500.00

\*/

ARQUIVO EXTERNO 182-Module\_part2.js

//MODULES:

//Aqui nós podemos ver como fazemos para que os nossos elementos sejam exportados, temos que usar a palavra reservada "exports" se usarmos essa palavra sobre um determinado elemento "mãe" todos os elementos dentro do mesmo escopo também serão exportados:

export class Pessoa {

    constructor(nome, cpf, rg, ){

        this.nome = nome

        this.cpf = cpf

        this.rg = rg

    }

    dadosPessoais(){ //Método gerado só para mostrar os dados pessoais da pessoa...

        console.log(`

        Nome: ${this.nome}

        CPF: ${this.cpf}

        RG: ${this.rg}

        `)

    }

}

//OBS: Se um elemento não tiver "export" ele não poderá ser acessado.

ARQUIVO package.js:

{

  "name": "javascript",

  "version": "1.0.0",

  "description": "",

  "main": "index.js",

  "dependencies": {

    "lodash": "^4.17.21"

  },

  "devDependencies": {},

  "scripts": {

    "test": "echo \"Error: no test specified\" && exit 1"

  },

  "author": "",

  "license": "ISC",

  "type": "module"

}