let saida;

//TIPO DE DADOS

    //NUMBER

    //String

    //Boolean

    //NaN

    saida = Number("Jamie");

    //underfined

    var nome;

    //console.log(nome)

    //null

    saida = null;

    if (saida){

        console.log("if");

    }else{

        console.log("else")

    } //o resul é vazio

   //array - arranjo

// do while - o tal do faça enquanto (Sempre vai rodar no mínimo uma vez)

let i = 0

do{

    console.log("executou o while: ",i)

    i++

}while(i < 5);

//loop FOR

//for sozinho --- let dentor de estruturas

for (let i =0;i < 5;i++){ //se mudar para let para var vc consegue acessar o valor por fora do for (por isso n se usa mais var! por dar problema)

   // console.log("Executou o for: ",i); //o que escrever fora n aparece e dá erro(console.log)

const valor = i;

}

//console.log("valor: ",valor);

//for dentro de for

for(let i = 0; i < 5; i++){//vai contar obviamente até o numero 4

//console.log("I: ",I);

    for(let i2 = 0; i2 < 5; i2++){

    }

    //console.log("I2",I2);

}

//ARRAY

    //DECLARAR

    let meuArray = ["Maça", "Pera","Arvore", 5]

    //ACESSAR

    saida = meuArray[0]

    //verificar tamanho

    saida = meuArray.length

    //transformar string em Array - método estático

    saida = Array.from("Jamie");

    //a saida fica ['j','a','m','i','e']

    //transformar string em Array de forma personalizada

    saida = "Jamie Reis".split(" ");

    //saida fica  ['Jamie', 'Reis']

    //atualizar

        //manualmente

        meuArray[0] = "Melancia"

        saida = meuArray;

        //inserindo no final

        meuArray.push("Abacate"); //aumenta o conteudo do array [Maça, Pera, 5 , Arvore, Abacate]

        //inserindo no começo

        meuArray.unshift("Limão");

    //apagar

        //apagar ultimo item

        meuArray.pop();

        //apagar primeiro item

        meuArray.shift();

        //apagar qualquer posição (posiçao, qtd)

        meuArray.splice( 1, 2 )

    //copiar array

    const novoArray =  meuArray.slice();

    saida = novoArray;

    //unir dois arrays

    const arr1 = [1,2];

    const arr2 = [3,4,5,6];

    let arrFinal = [];

    //com for

/\*for (let index = 0; index < arr1.length; index++) {

   arrFinal.push(arr1[index])

}for (let index = 0; index < arr2.length; index++) {

    arrFinal.push(arr2[index])  //push é empurrando o conteudo de um para dentro do outro array

 }

 saida = arrFinal;

\*/

    //com SPREAD -- ajuda a espalhar o conteudo do array

    arrFinal = [...arr1,...arr2];

    saida = arrFinal;

    //com concat()

    arrFinal = arr1.concat(arr2)

    saida = arrFinal;

//HOFS --funções de ordem superior

 /\*/   ///forEach/////

 -> Itera valores do array sem retonar nada

 -> vai ate o final

 -> [e,i,a]

 ->(e) Não modifica o original

 -> (a) Modifica o original

\*/

 const fruits = ["apple","orange","cherry"]

fruits.forEach(function(e,i,a){ //forEach n retorna nada, tem q ser depois e dentro da propria fruits

    console.log("alterando: ",e);

 //ele meio que itera as fuções dos valores que estão dentro de fuction

 //callbackfn - fn no final significa q espera uma função

// e = "novo"//a cada iterada ele vai trocar os valores

 //precisa de um return sempre!!!

a[i] = "novo"

 return true;

});

console.log("fruits: ", fruits);

//essa parte n é esperada

//console.log("saida: ", saida)

//console.log('fruits: ', fruits)

/\*

///////// find ////////////

->Vai ate onde a função retorna um valor TRUE

->Find vai retorna o VALOR ATUAL da iteração no momento que o retorno foi TRUE

->Params (e,i,a)

\*/

const ages = [3, 20, 18, 18];

saida = ages.find (function(e,i,a){ //retorna o valor do i (20)

return e > 18

});

/\*

/////////some//////////

-> Vai ate a função retonar um valor TRUE

-> Some: Retorna TRUE caso exista algum retorno positivo e FALSE se não houver

->Params (e,i,a)

\*/

//const ages = [3, 10, 18, 20];

saida = ages.some

/\*

FILTER()

- Devolve um novo array contendo somente os itens que retornam TRUE

- Não modifica o original

- params (c,i,a)

\*/

const numbers = [1,2,3,4,5];

saida = numbers.filter(function(e,i,a){

 return e < 3

})

/\* MAP()

-Itera todos os itens do array ()

- só itera o array original

- Constrói um novo array com base no retorno de cada chamada. (Voce pode retornar qualquer coisa)

- params(c,i,a)

\*/

saida = numbers.map(function(e,i,a){ return "Jamie" + e}) //retorna os dados mas reformulando-os

saida = numbers

//com hof REDUCE()

//aSSIM COMO O MAP() PODE RETORNAR QUALQUER COISA

//POSSUI UM PARAMENTRO QUE GUARDA A INFORMAÇÃO PASSADA PARA ELE NO LOOP ANTERIOR.

/\*

/////// SORT ///////////

-> Reordena o conteúdo do array original baseando-se no valor de retorno da função.

--> Esse valor de retorno é gerado baseado em 3 possibilidades de retorno (negativo, positivo, zero) e devem ser utilizados os 2 parametros fornecidos para gerar uma dessas 3 respostas.

--> Alem de modificar o array de origem ele tbm retorna um novo array

\*/