<LAB365>

JavaScript Básico - Parte 1

AGENDA | M1S03 - A1

- O que é?
- Tipos de dados
- Variáveis
- Operadores Aritméticos

JAVASCRIPT

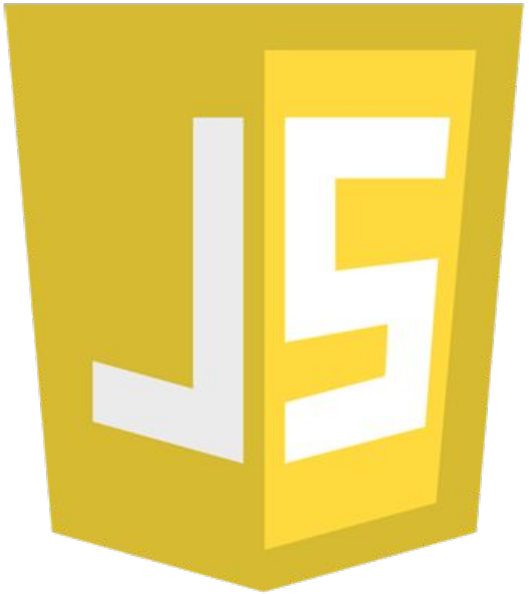
É uma linguagem de programação de alto nível, dinâmica e interpretada, amplamente utilizada para adicionar **interatividade e comportamento dinâmico a páginas web**.

Criada originalmente por Brendan Eich em 1995, ela se tornou uma das linguagens mais populares e essenciais para o desenvolvimento web, funcionando tanto no lado do cliente (front-end) quanto no lado do servidor (back-end).

JAVASCRIPT

A execução do JavaScript ocorre principalmente em dois ambientes: no **navegador** (front-end) e no servidor (back-end), com o uso do Node.js. Em ambos os casos, o JavaScript é uma linguagem interpretada, o que significa que o código é executado linha por linha, sem necessidade de compilação prévia.

No navegador, o JavaScript é executado no motor de JavaScript integrado (como o V8 no Chrome), permitindo interatividade em páginas web, como manipulação do DOM, respostas a eventos do usuário e requisições assíncronas (AJAX).



TIPOS DE DADOS

Em JavaScript, os tipos de dados são categorizados em dois grupos principais: tipos primitivos e tipos de referência (objetos). Esses tipos definem o tipo de valor que uma variável pode armazenar e como ele pode ser manipulado.

Os tipos primitivos são valores imutáveis (não podem ser alterados diretamente) e são armazenados diretamente na memória.

Os tipos de referência são mutáveis e armazenam referências a valores na memória.

TIPOS DE DADOS

Tipos primitivos

- string: Representa uma sequência de caracteres.
- number: Representa números, tanto inteiros quanto decimais.
- boolean: Representa um valor lógico: true (verdadeiro) ou false (falso).
- undefined: Representa uma variável que foi declarada, mas não teve um valor atribuído.
- null: Representa a ausência intencional de valor.
- bigint: Representa números inteiros maiores do que o limite do tipo number.
- symbol: Representa um valor único e imutável, frequentemente usado como identificador de propriedades de objetos.

Tipos de referência

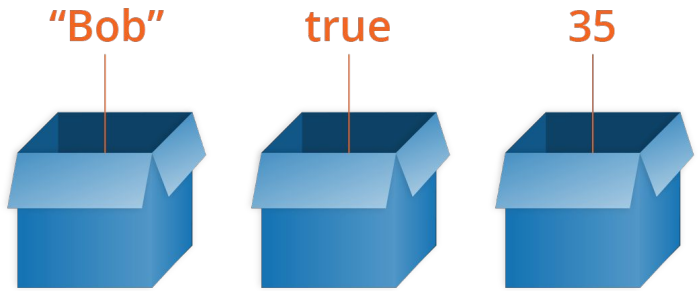
- object: Estrutura de dados que armazena pares de chave-valor.
- array: Uma lista ordenada de valores.
- function: Blocos de código reutilizáveis que podem ser invocados.

VARIÁVEIS

Variáveis em JavaScript são **espaços na memória** usados para armazenar dados que podem ser **acessados e manipulados** ao longo do código.

Elas são essenciais para **guardar valores**, como números, textos, objetos, ou qualquer outro tipo de dado, e reutilizá-los posteriormente.

Em JavaScript, existem três palavras-chave para declarar variáveis: **var, let e const**. Cada uma tem suas particularidades em termos de escopo e mutabilidade.



TIPOS DE VARIÁVEIS

- var: Era a forma original de declarar variáveis no JavaScript, **pode ser redeclarada e atualizada**.
- let: Introduzida no ES6 (ECMAScript 2015), pode ser atualizada, mas **não redeclarada no mesmo escopo**.
- const: Também introduzida no ES6, **não pode ser redeclarada nem atualizada** (imutável).

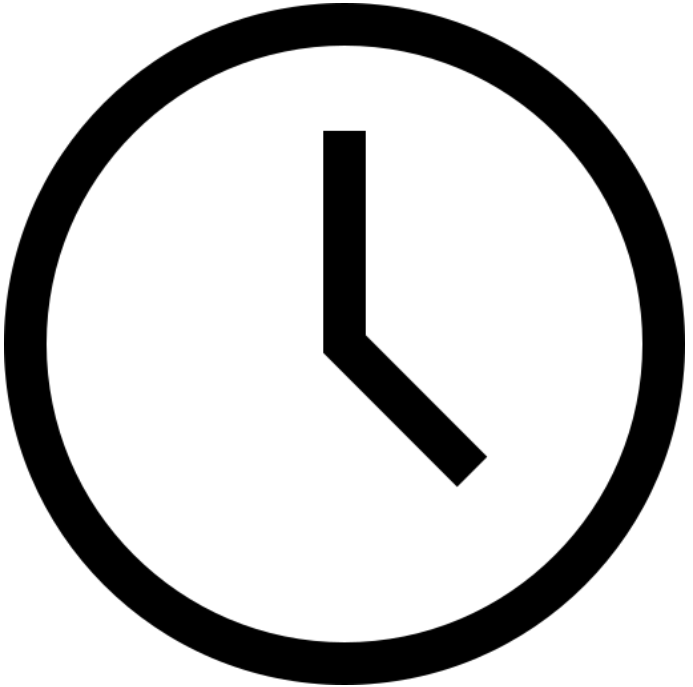
```
let nome = "Maria"; // Variável mutável
const idade = 30;   // Variável imutável

console.log(nome);  // "Maria"
console.log(idade); // 30

nome = "João";      // Atualizando o valor de 'nome'
console.log(nome);  // "João"

// idade = 31;      // Isso causaria um erro, pois 'const' não pode ser atualizada.
```

INTERVALO!

Finalizamos o nosso primeiro período de hoje. Que tal descansar um pouco?!

Nos vemos em 20 minutos.

Início: 20:25
Retorno: 20:45



OPERADORES ARITMÉTICOS

Os operadores aritméticos são usados para realizar operações **matemáticas básicas**.

Utilizando **variáveis e os operadores** nós podemos estar desenvolvendo códigos para lidar com os mais diversos tipos de cálculos matemáticos, desde uma simples soma até a realização do cálculo da fórmula de bhaskara.

OPERADORES ARITMÉTICOS

Operador	Descrição	Exemplo	Resultado
`+`	Adição	`10 + 5`	`15`
`-`	Subtração	`10 - 5`	`5`
`*`	Multiplicação	`10 * 5`	`50`
`/`	Divisão	`10 / 5`	`2`
`%`	Módulo (resto da divisão)	`10 % 3`	`1`
`**`	Exponenciação	`2 ** 3`	`8`

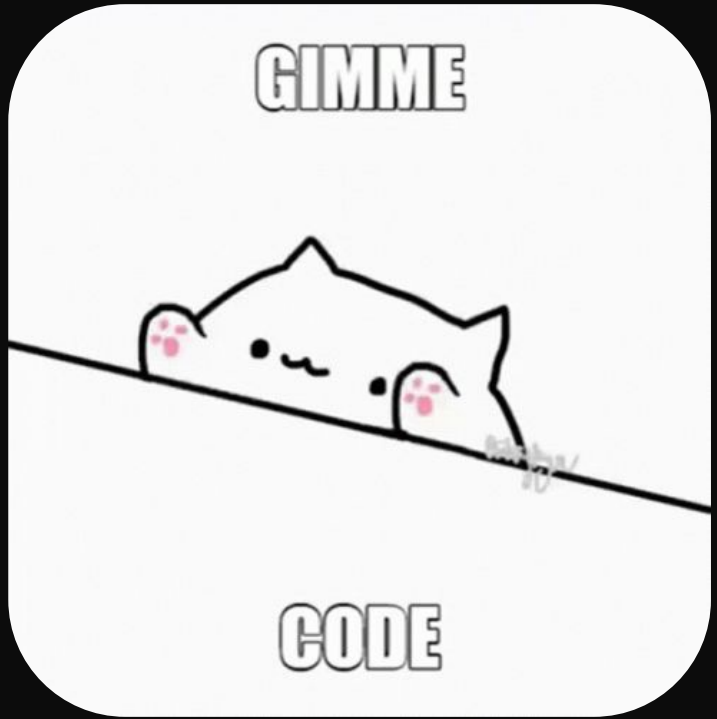
EXEMPLO DE CÓDIGO

```
let a = 10;
let b = 5;

console.log(a + b); // 15 (adição)
console.log(a - b); // 5 (subtração)
console.log(a * b); // 50 (multiplicação)
console.log(a / b); // 2 (divisão)
console.log(a % b); // 0 (resto da divisão)
console.log(a ** b); // 100000 (exponenciação)
```

TREINANDO NOSSAS HABILIDADES!

Como forma de treinar nosso conhecimento adquirido em Javascript, vamos desenvolver dois arquivos, cada uma para atender a necessidade abaixo:

- Cálculo de velocidade média (fórmula de velocidade média).
- Cálculo da fórmula de Bhaskara(uma fórmula para x’ e outra para x”).



<LAB365>

<LAB365>

JavaScript Básico - Parte 2

<LAB365> SENAI

AGENDA | M1S03 - A2

- Operadores lógicos
- Operadores de comparação
- Estruturas condicionais
  - If
  - Else
  - Else if
  - Switch

OPERADORES LÓGICOS

os operadores lógicos são usados para realizar **operações sobre valores booleanos (true ou false)**. Eles são frequentemente utilizados em estruturas condicionais e loops para tomar decisões com base em múltiplas condições.

Operador	Nome	Uso Comum
`&&`	AND	Verificar se **todas** as condições são verdadeiras.
`||`	OR	Verificar se **pelo menos uma** condição é verdadeira.
`!`	NOT	Inverter o valor de uma condição.

OPERADORES LÓGICOS

E			
	Vermelho	Verde	Resultado
Cor	Verdadeiro	Falso	Falso
Cor	Falso	Verdadeiro	Falso
Cor	**Verdadeiro**	**Verdadeiro**	**Verdadeiro**

OPERADORES LÓGICOS

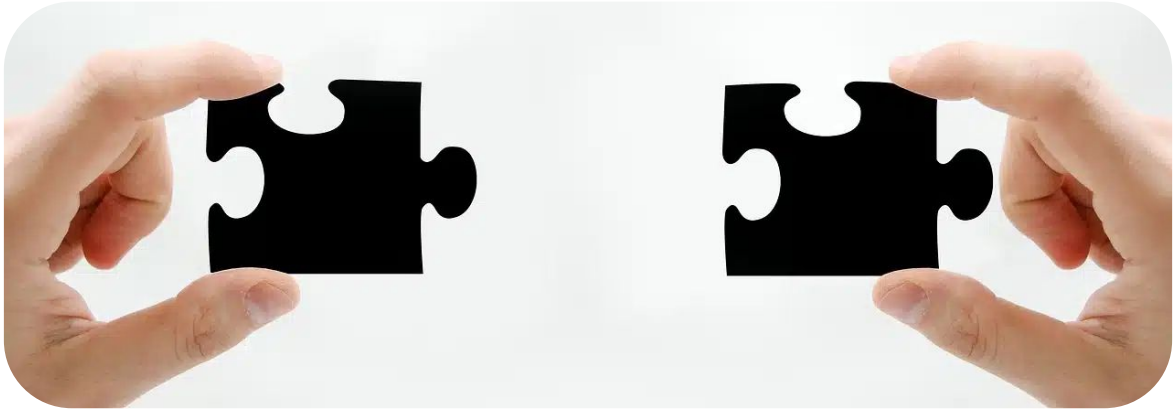
OU			
	Vermelho	Verde	Resultado
Cor	**Verdadeiro**	**Falso**	**Verdadeiro**
Cor	**Falso**	**Verdadeiro**	**Verdadeiro**
Cor	**Verdadeiro**	**Verdadeiro**	**Verdadeiro**
Cor	Falso	Falso	Falso

OPERADORES LÓGICOS

NEGAÇÃO		
	Vermelho	Resultado
Cor	Verdadeiro	Falso
Cor	**Falso**	**Verdadeiro**
Cor	Verdadeiro	Falso
Cor	**Falso**	**Verdadeiro**

OPERADORES DE COMPARAÇÃO

Os operadores de comparação são usados para comparar dois valores e retornar um valor booleano (true ou false). Eles são fundamentais para tomar decisões em estruturas condicionais, como if, while, e for.



OPERADORES DE COMPARAÇÃO

Operador	Descrição	Exemplo	Resultado
`==`	Igualdade (valor)	`10 == "10"`	`true`
`===`	Igualdade estrita (valor e tipo)	`10 === "10"`	`false`
`!=`	Diferença (valor)	`10 != "10"`	`false`
`!==`	Diferença estrita (valor e tipo)	`10 !== "10"`	`true`
`>`	Maior que	`10 > 5`	`true`
`<`	Menor que	`10 < 5`	`false`
`>=`	Maior ou igual a	`10 >= 10`	`true`
`<=`	Menor ou igual a	`10 <= 5`	`false`

IF E ELSE

Em programação, if e else são estruturas condicionais que permitem que o programa tome decisões com base em condições específicas. Essas estruturas são como "desvios" no fluxo do código: dependendo do resultado de uma expressão ou condição, o programa pode seguir caminhos diferentes.

IF E ELSE

O if é usado para verificar uma condição. Se essa condição for verdadeira, o bloco de código associado ao if é executado.

O else é opcional e funciona como um "plano B". Se a condição do if for falsa, o bloco de código associado ao else é executado.

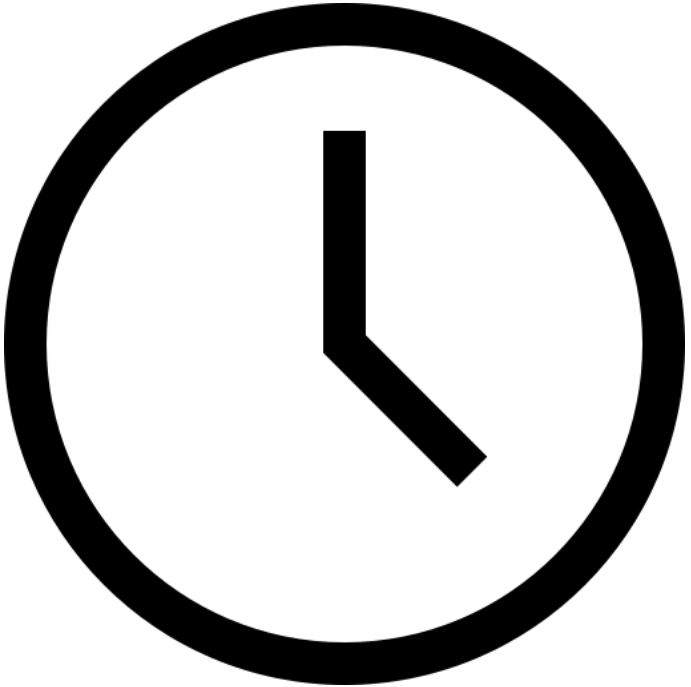
```
if (condição) {
    // Código a ser executado se a condição for verdadeira
} else {
    // Código a ser executado se a condição for falsa
}
```

INTERVALO!

Finalizamos o nosso primeiro período de hoje. Que tal descansar um pouco?!

Nos vemos em 20 minutos.

Início: 20:30
Retorno: 20:50



ELSE IF

A estrutura else if permite testar múltiplas condições em sequência. Se a condição do if for falsa, o else if é verificado, e assim por diante.

```
if (condição1) {
    // Código a ser executado se condição1 for verdadeira
} else if (condição2) {
    // Código a ser executado se condição2 for verdadeira
} else if (condição3) {
    // Código a ser executado se condição3 for verdadeira
} else {
    // Código a ser executado se nenhuma condição for verdadeira
}
```

EXEMPLO DE CÓDIGO

```
let nota = 75;

if (nota >= 90) {
    console.log("Aprovado com mérito!");
} else if (nota >= 60) {
    console.log("Aprovado.");
} else {
    console.log("Reprovado.");
}
// Resultado: "Aprovado."
```

SWITCH

O switch é uma estrutura de controle usada para tomar decisões com base no valor de uma variável ou expressão.

Ele é uma alternativa ao uso de vários if e else if quando você tem muitas condições para verificar.

```
switch (expressao) {
    case valor1:
        // Código a ser executado se a expressão for igual a valor1
        break;
    case valor2:
        // Código a ser executado se a expressão for igual a valor2
        break;
    // Você pode ter quantos "case" quiser
    default:
        // Código a ser executado se nenhum dos casos for correspondido
}
```

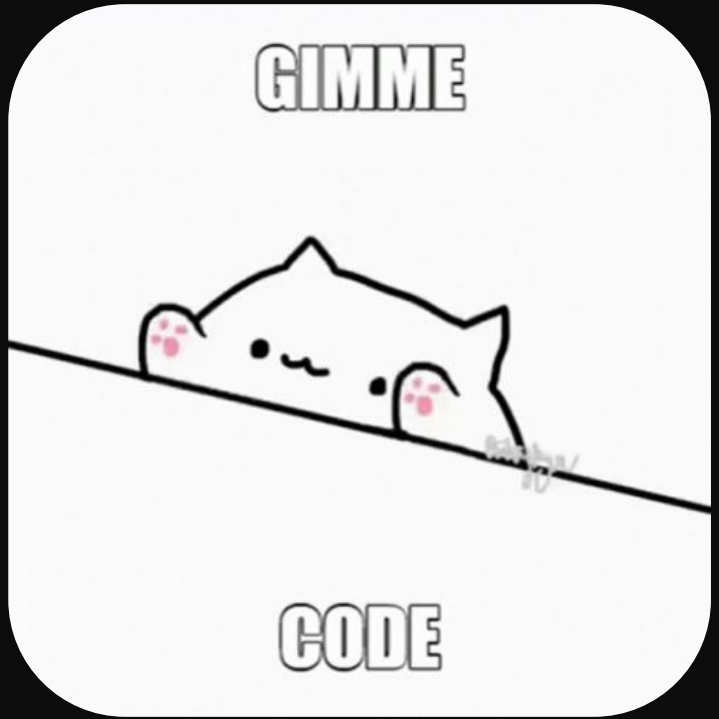
EXEMPLO DE CÓDIGO

```
let diaDaSemana = 3;
let nomeDoDia;

switch (diaDaSemana) {
  case 1:
    nomeDoDia = "Domingo";
    break;
  case 2:
    nomeDoDia = "Segunda-feira";
    break;
  case 3:
    nomeDoDia = "Terça-feira";
    break;
  case 4:
    nomeDoDia = "Quarta-feira";
    break;
  case 5:
    nomeDoDia = "Quinta-feira";
    break;
  case 6:
    nomeDoDia = "Sexta-feira";
    break;
  case 7:
    nomeDoDia = "Sábado";
    break;
  default:
    nomeDoDia = "Dia inválido";
}

console.log(nomeDoDia); // Saída: "Terça-feira"
```

TREINANDO NOSSAS HABILIDADES!

Como forma de treinar nosso conhecimento adquirido em Javascript, vamos desenvolver dois arquivos, cada uma para atender a necessidade abaixo:

- Verificação de maioridade (use nome de idade de variáveis).
- Verificação de valor em excursão, se a pessoa tiver menos de 5 anos é gratuita, se tiver menos de 12 anos o valor é R$10, se for menor de 18 o valor é 10 + metade da idade da pessoa e acima de 18 o valor é 10 + a idade da pessoa. (use nome e idade de variáveis)



<LAB365>

<LAB365>

JavaScript Básico - Parte 3
+ Revisão

<LAB365> SENAI

AGENDA | M1S03 - A3

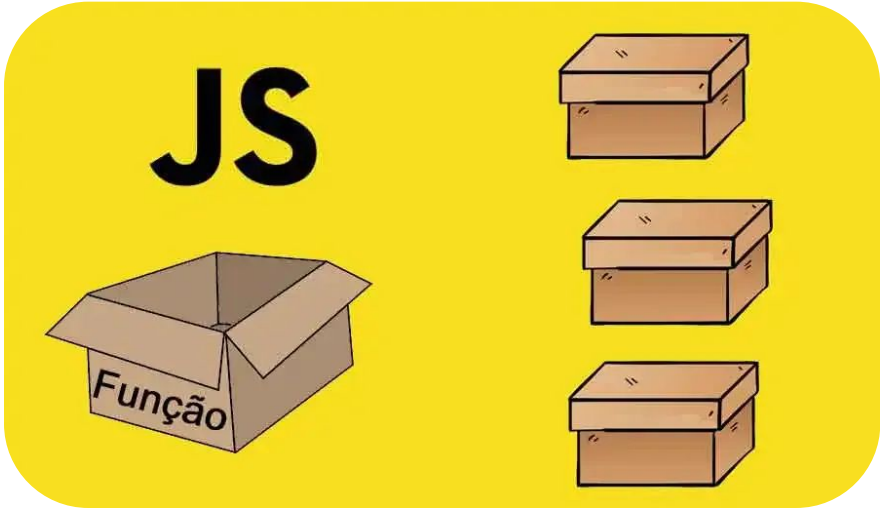
- Funções
- Revisão
- Correções

FUNÇÕES

Vamos falar sobre funções no JavaScript, que são um dos conceitos mais importantes e poderosos da linguagem.

Funções permitem que você **encapsule blocos de código** para **reutilização**, organização e execução sob demanda.

Uma função é um bloco de código que pode ser **definido uma vez** e **executado quantas vezes você quiser**. Ela pode receber parâmetros (dados de entrada) e retornar um valor (dado de saída).



FUNÇÕES

Existem várias maneiras de definir funções em JavaScript.

- Função declarada: o método mais clássico dentro da programação de se trabalhar com funções.

```
function nomeDaFuncao(parametro1, parametro2) {
    // Código a ser executado
    return resultado; // Opcional
}
```

- Função expressa (anônima): a função é atribuída a uma variável.

```
const nomeDaFuncao = function(parametro1, parametro2) {
    // Código a ser executado
    return resultado; // Opcional
};
```

EXEMPLO DE CÓDIGO

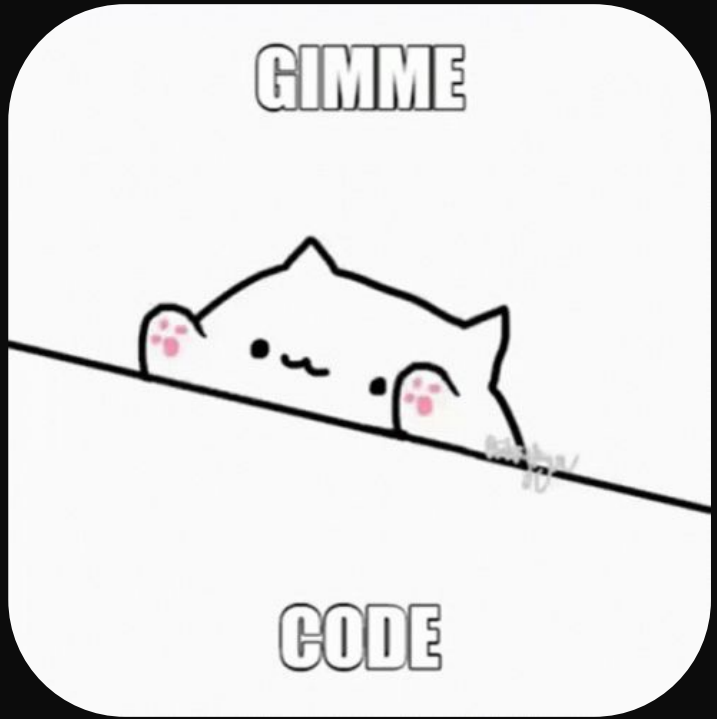
```
function soma(a, b) {
    return a + b;
}

console.log(soma(2, 3)); // Saída: 5

// ----- //

const multiplicacao = function(a, b) {
  return a * b;
};

console.log(multiplicacao(2, 3)); // Saída: 6
```

TREINANDO NOSSAS HABILIDADES!

Como forma de treinar nosso conhecimento adquirido, vocês devem criar o código que fará a verificação se a pessoa pode entrar com o pedido de aposentadoria.

Lembrando que para isso temos de considerar a idade da pessoa, sexo (masculino ou feminino) e o tempo de contribuição. Para verificar se a entrada na aposentadoria será iniciado ou negada.

Estruture o código da forma que achar melhor.

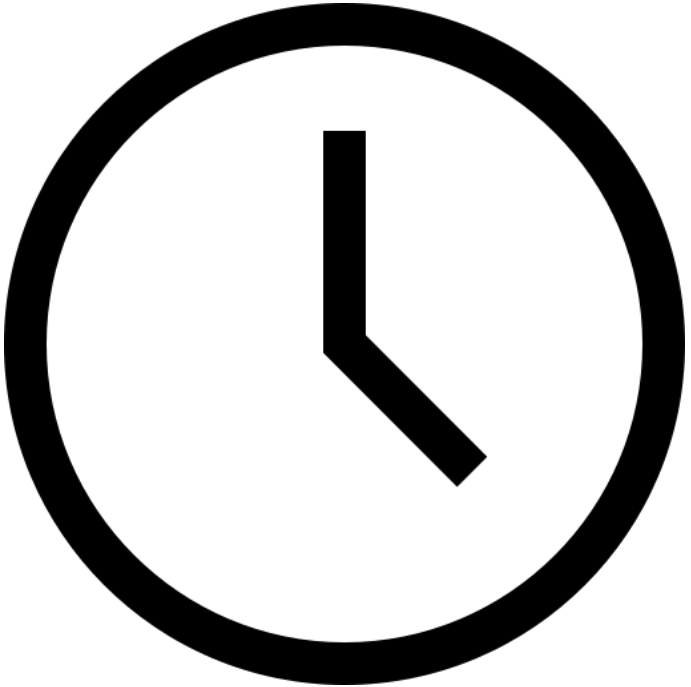


INTERVALO!

Finalizamos o nosso primeiro período de hoje. Que tal descansar um pouco?!

Nos vemos em 20 minutos.

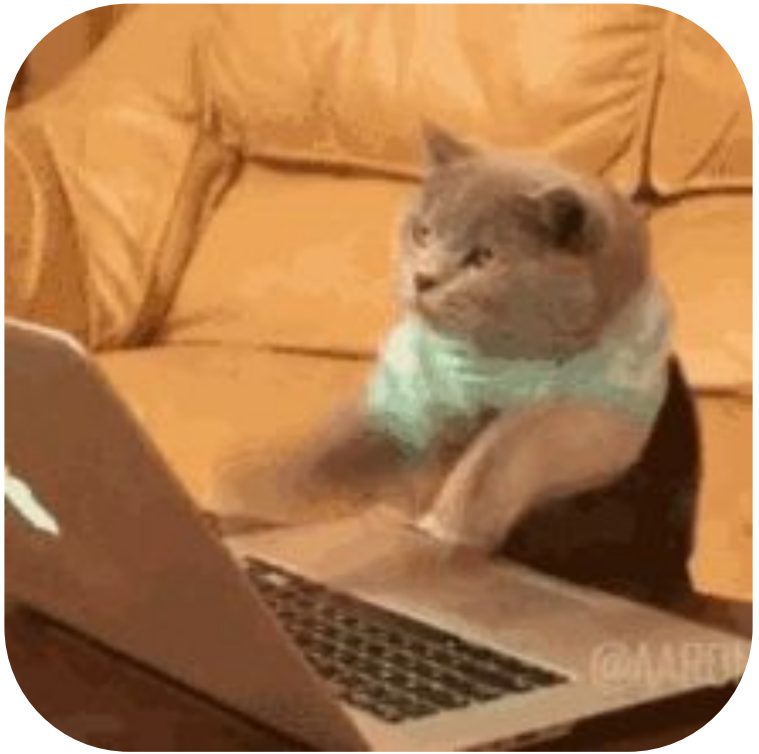
Início: 20:20
Retorno: 20:40



REVISÃO E CORREÇÕES

Nossa revisão será programando!

- JavaScript
  - Tipos de dados
  - Variáveis
  - Operadores Aritméticos
  - Operadores Lógicos
  - Operadores de comparação
  - Estruturas Condicionais
  - Funções



AVALIAÇÃO DOCENTE

O que você está achando das minhas aulas neste conteúdo?

Clique aqui ou escaneie o QRCode ao lado para avaliar minha aula.

Sinta-se à vontade para fornecer uma avaliação sempre que achar necessário.

<LAB365>