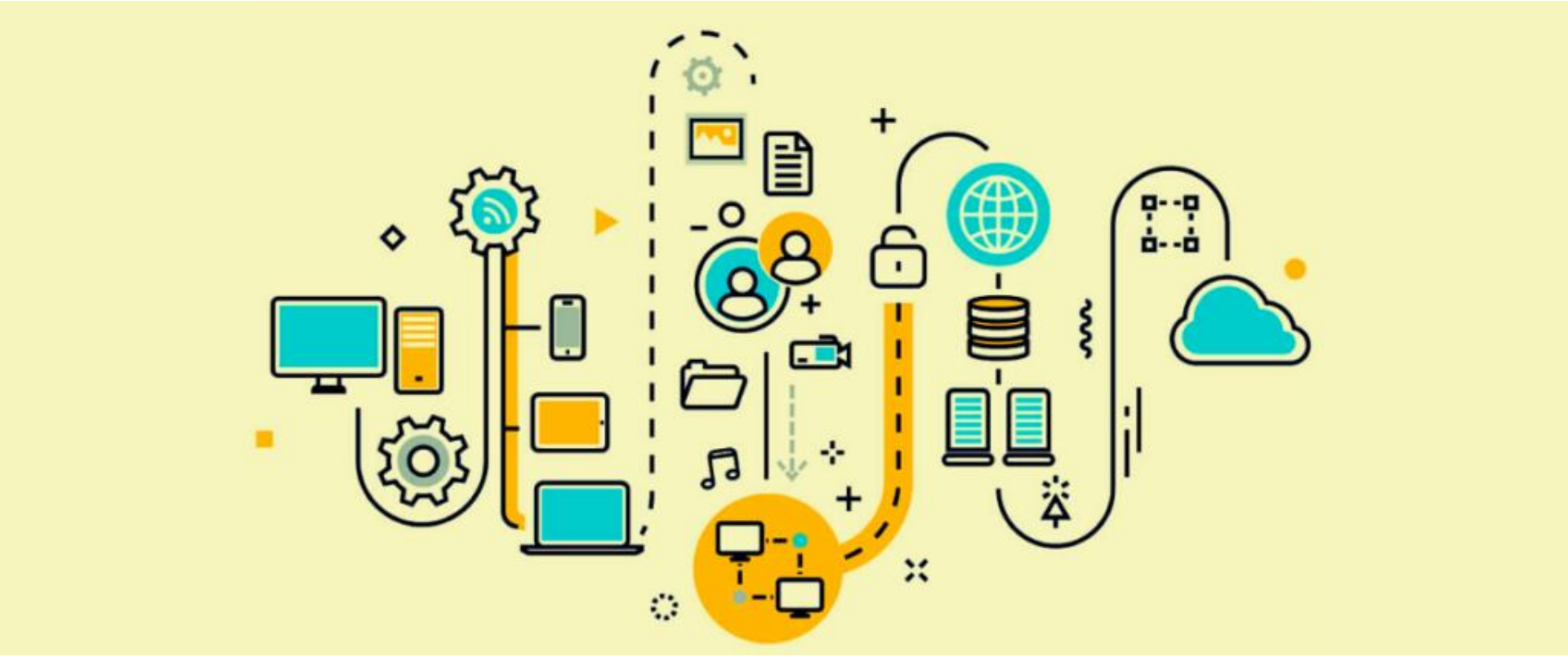


# ROTEIRO

## Parte 1

- Recomendações
- Reflexões
- Conceito de Algoritmo
- Cuidados ao Escrever Um Algoritmo
- Linguagens de Programação
- Exemplo de Algoritmo

## Parte 2

- Conhecendo o Portugol Studio
- Tipos de Variáveis
- Operadores Aritméticos, Relacionais e Lógicos

## Parte 3

- Exemplo de Programa no Portugol Studio
- Exercícios

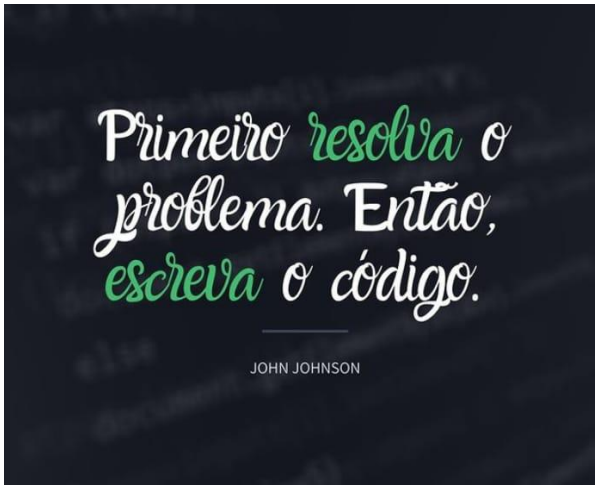
# REFLEXÕES

"POR MAIS BRILHANTE QUE SEJA A CAPACIDADE, SEM TREINAMENTO, NÃO HÁ EVOLUÇÃO"

# ALGORITMO



## COMO CONSTRUIR ALGORITMOS




- Compreender o problema: definir qual o objetivo do algoritmo.
- Definir as informações de entrada: que informações precisamos obter do usuário.
- Definir o processamento: que cálculos devemos efetuar. É neste momento que os dados obtidos pela entrada serão transformados em informação útil para o usuário.
- Definir as informações de saída: que informações devemos fornecer ao usuário como resultado do processamento efetuado.

# CUIDADOS AO ESCREVER UM ALGORITMO

1. A ordem lógica da execução das tarefas é importante.
2. Todo algoritmo tem início e fim.
3. Um algoritmo tem que ser completo.
4. Um algoritmo deve ter um alto índice de detalhamento.
5. Cada tarefa ou etapa é chamada de instrução.

# LÓGICA DE PROGRAMAÇÃO



Para falar, o ser humano antes pensa, organiza o pensamento em forma de sentenças e reproduz através da fala. Esta fala segue alguma **linguagem** que consequentemente segue algumas regras e padrões

Da mesma forma, os **programas de computadores** devem antes ser **entendidos, analisados, pensados e organizados.**

O Algoritmo possui a lógica de tudo o que o programa deverá fazer.

**A Lógica é a forma correta de se pensar, é também uma forma de organizar as ideias.**

# EXEMPLO DE ALGORITMO



**Exemplo de algoritmo**

Imagine o seguinte problema: calcular a media final dos alunos da 3a serie.

Os alunos realizaram quatro provas: P1, P2, P3 e P4.

Onde: Media Final = (P1 + P2 + P3 + P4) / 4

Para montar o algoritmo proposto, faremos três perguntas:

a) Quais são os dados de entrada?

R: Os dados de entrada são P1, P2, P3 e P4.

b) Qual será o processamento a ser utilizado?

R: O procedimento será somar todos os dados de entrada e dividi-los por 4 (quatro) (P1 + P2 + P3 + P4) / 4.

c) Quais serão os dados de saída?

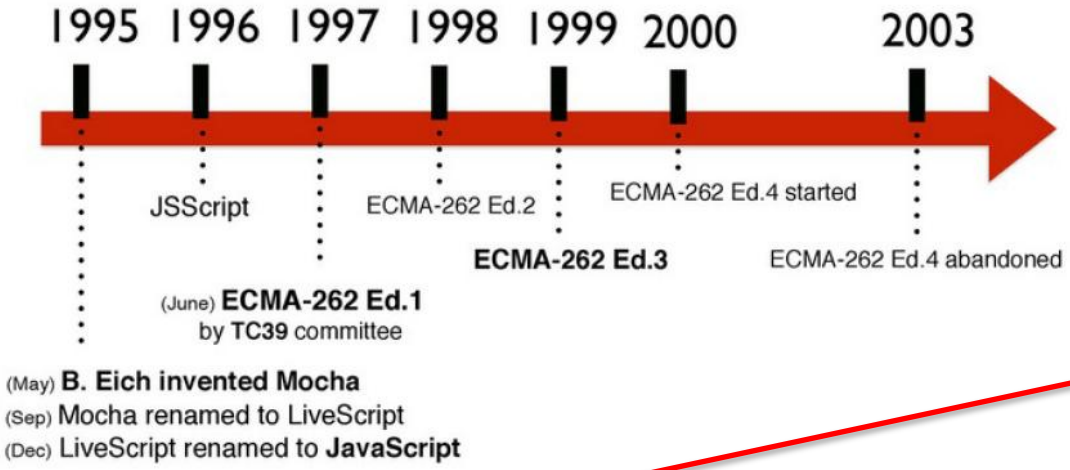
R: O dado de saída será a média final.

# Dúvidas ??

# NOSSA FERRAMENTA DE TRABALHO


# JAVASCRIPT

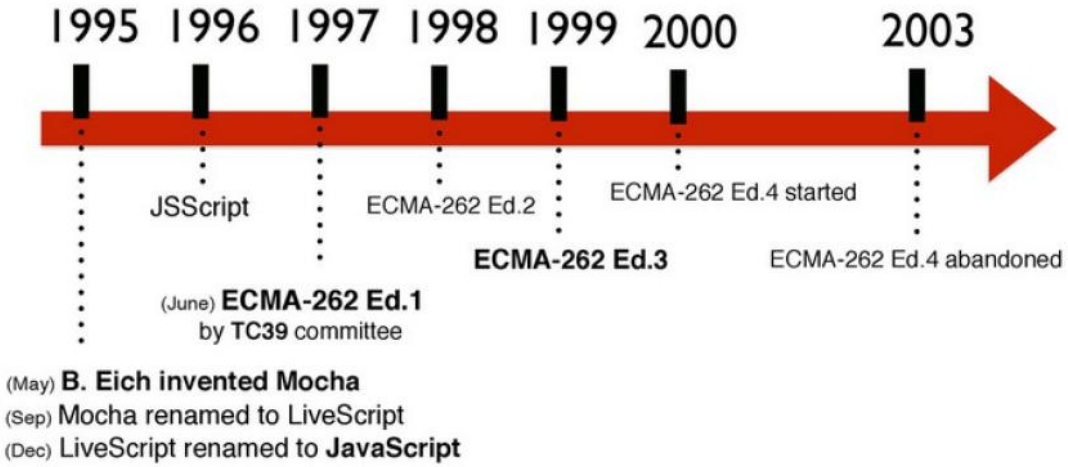
# HISTÓRIA




Em **1995**, Brendan Eich (ex-CEO da Mozilla) desenvolveu o que seria a primeira versão JavaScript para o navegador Netscape Navigator.
Naquela época era chamado **Mocha** e mais tarde foi renomeado para **LiveScript**.

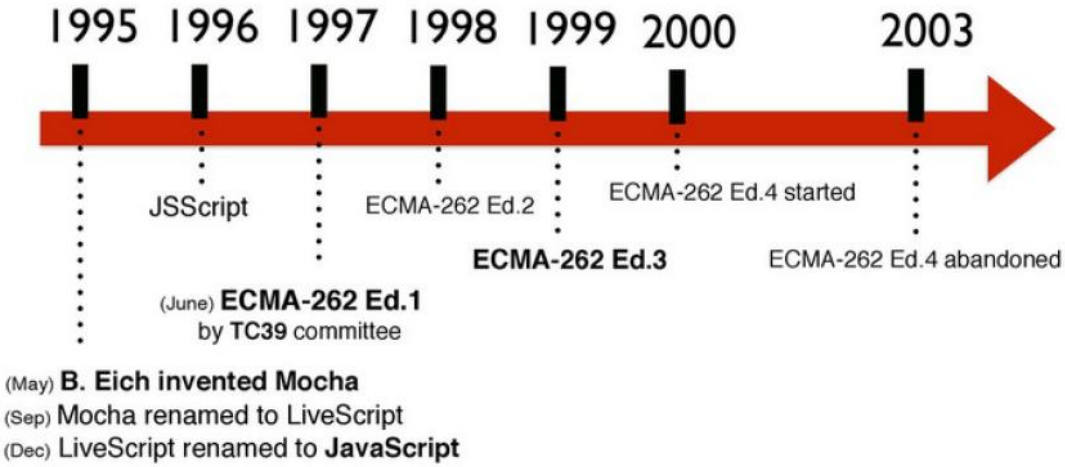
O nome **JavaScript** foi dado porque o Netscape adicionou suporte a Java em seu navegador e foi uma tecnologia muito popular na época

# HISTÓRIA

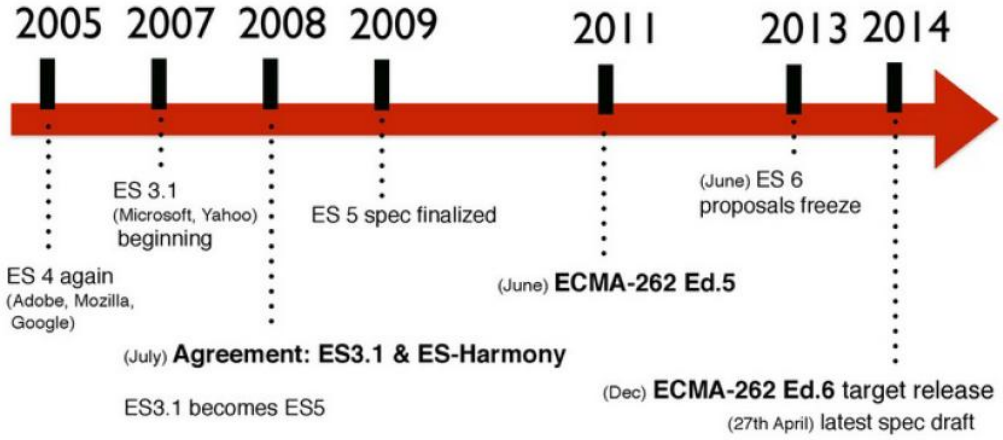



Em **1997**, um comitê (chamado TC39) foi criado para criar um padrão JavaScript pelo Associação Europeia de Fabricantes de Computadores, **ECMA**. Nesta comissão, o padrão do **DOM**, Document Object Model para, desta forma, evitar incompatibilidades entre navegadores.
É a partir de então que o os padrões JavaScript começarem a ser ditados pelo **ECMAScript**.

# HISTÓRIA




Em **1999**, a versão 3 do JavaScript foi padronizada e permaneceu em vigor até recentemente. Houve algumas tentativas de lançar uma versão 4, mas a que finalmente padronizado e ainda até agora é a versão 5 do **ECMAScript**, aprovado em **2011**. Em junho de **2013**, o rascunho da versão 6 parou, mas em dezembro de **2014** foi finalmente aprovado e padronizado em julho de **2015**.



# HISTÓRIA

O JavaScript foi projetado para adicionar efeitos e animações a sites da Web, mas foi evoluindo muito ao longo dos anos, tornando-se uma linguagem polivalente. É a partir de **2005**, com a chegada do GMail e seu uso de tecnologia AJAX, JavaScript Assíncrono e XML (graças ao objeto XMLHttpRequest criado pela Microsoft para Internet Explorer 5.0), que lançou sua popularidade.

# O QUÊ IREMOS APRENDER

Aqui você encontra por disciplina alguns recursos que irão te auxiliar no aprendizado

1. JavaScript
2. TypeScript
3. Node.Js
4. React Native
5. Firebase
6. SQL
7. Git e GitHub

# VARIÁVEIS

Variáveis em programas são espaços de memória reservados para armazenar valores que podem ser modificados durante a execução do programa. Elas são utilizadas para armazenar informações temporárias ou valores que precisam ser manipulados e utilizados durante a execução do código.

As variáveis têm nomes associados a elas, que são usados para referenciá-las e acessar os dados armazenados. Cada variável possui um tipo de dado que define o tipo de informação que pode ser armazenado nela, como números inteiros, números de ponto flutuante (números reais), caracteres, strings (sequências de caracteres), entre outros.

# VARIÁVEIS

Em resumo, variáveis são nada mais nada menos do que

**RÓTULOS PARA ENDEREÇOS DE MEMÓRIA**

```
// Exemplo de pseudo-código
inteiro idade;
real salario;
caractere letra;
texto nome;

idade = 30;
salario = 2500.50;
letra = 'A';
nome = "João";

escrever("A idade é: ", idade);
escrever("O salário é: ", salario);
escrever("A letra é: ", letra);
escrever("O nome é: ", nome);
```

# TIPOS DE VARIÁVEIS

Os tipos mais utilizados no JavaScript são:

- number
- boolean
- string
- undefined

**NUMBER** - Utilizados para armazenar valores numéricos. São utilizados para contagens, cálculos e comparações. Vamos á prática.

**SPECIAL NUMBERS** – São considerados números mas não são números.

**Infinity** - operações matemáticas que dão errado e geram um número gigante.

**NaN** - Tipo uma tentativa de somar algo num String

Vamos á prática.

**STRINGS -** As variáveis de cadeia de caracteres ou chamadas Strings, armazenam caracteres ou palavras. Elas são delimitadas por aspas simples ou duplas. Vamos á prática.

# TIPOS DE VARIÁVEIS



## O que mais a gente pode fazer com Strings

- A "\" pode dar um "escape" na String e isso permite gerar efeitos especiais.

\n - pula linha
\t - cria um caracter de tabulação

- Para inserir uma variável na String a gente deve começar o String com aspas invertidas `, tipo:

```
console.log(`O valor de 2 + 2 é : ${2 + 2}`);
```

- A gente também pode concatenar Strings, tipo:

```
console.log("O aluno " + "está " + "APROVADO");
```

# TIPOS DE VARIÁVEIS

**BOOLEAN** - Esse tipo de dados armazena um bit que indica verdadeiro ou falso. Os valores booleanos são usados para indicar estados. Por exemplo, atribuímos a uma variável o estado false no início de uma operação e a alteramos para true no final da operação. Em seguida, realizamos a verificação necessária.

Vamos á prática.

# OPERADORES



## OPERADORES ARITMÉTICOS

| Operador | Significado |
|---|---|
| + | Soma |
| – | Subtração |
| * | Multiplicação |
| / | Divisão (normal) |
| % | Módulo |
| ++ | Incremento |
| — | Decremento |

# OPERADORES



## OPERADORES RELACIONAIS

| Operador | Operação | Exemplo |
|---|---|---|
| > | Maior que | (a > b) |
| < | Menor que | (a < b) |
| >= | Maior ou igual a | (a >= b) |
| <= | Menor ou igual a | (a <= b) |
| == | Igual a | (a == b) |
| != | Diferente de | (a !== b) |
| === | Idêntico a | (a === b) |
| !== | Não idêntico a | (a !== b) |
| && | E/and | (a && b) |
| ll | Ou/or | (a ll b) |

# OPERADORES



## OPERADORES DE ATRIBUIÇÃO

| Operador | Descrição | Exemplo |
|---|---|---|
| = | Atribuição | C = A + B atribui o valor de A + B em C |
| += | Atribuição de soma | C += A equivale a C = C + A |
| -= | Atribuição de subtração | C -= A equivale a C = C – A |
| *= | Atribuição de multiplicação | C *= A equivale a C = C * A |
| /= | Atribuição de divisão | C /= A equivale a C = C / A |
| %= | Atribuição de resto | C %= A equivale a C = C % A |

# OPERADORES



## OPERADORES LÓGICOS

| Operador | Operação | Exemplo |
|---|---|---|
| && | E/and | (a && b) |
| ll | Ou/or | (a ll b) |

# DIFERENÇA ENTRE “==“ E “===“

Se usarmos o operador “==” em uma condição, saberemos que ela só será verdadeira se o **valor da esquerda for o mesmo valor da direita**, por exemplo:

```
if (true == “true”) //retorna true

if (10 == “10”) //retorna true

if (true == true) //retorna true

if (10 == 10) //retorna true
```

Mas suponha que seja preciso fazer essa validação tendo certeza de que o número da direita é realmente um número, e não apenas possui o mesmo caractere. O que fazemos?
É nessa hora que entra o símbolo de “idêntico a” (===). Ele não só **compara os valores** dos dois lados da equação, como também verifica **se eles são do mesmo tipo**. Por exemplo:

```
if (true === “true”) //retorna false

if (10 === “10”) //retorna false

if (true === true) //retorna true

if (10 === 10) //retorna true
```

# DECLARANDO VARIÁVEIS

Existem formas de declarar variáveis no JavaScript. Já utilizamos algumas, mas vale a pena ver todas.

```
var nomeAluno = “Pedro Silveira”;
console.log(nomeAluno);

const sobrenomeAluno = “e Souza”;
console.log(nomeAluno + “ “  + sobrenomeAluno);

let nomeFruta = “Laranja”;
console.log(nomeFrutarenomeAluno);
```

? Qual a diferença entre a utilização de **Let,Var e Const ?**

var (escopo global) - escopo fora do bloco.
let (escopo local)  - escopo restrito ao bloco.
const (não permitindo reatribuição e nem redeclaração)

# CONVENÇÃO DE NOMES DE VARIÁVEIS



A convenção de nomes de variáveis é uma prática fundamental em programação que consiste em seguir regras e padrões ao nomear variáveis em um código-fonte. Embora não afete diretamente o funcionamento do programa, a importância da convenção de nomes reside em tornar o código mais legível, compreensível e sustentável. Algumas razões pelas quais a convenção de nomes é relevante incluem:

- **Legibilidade e compreensão**
- **Manutenção e colaboração**
- **Evita erros e conflitos**
- **Consistência**
- **Boas práticas da comunidade**

# CONVENÇÃO DE NOMES DE VARIÁVEIS



Não pode começar com um número (let 2teste);

Mas pode terminar com número (let teste9 = “testando”)

Pode ter $ ou _,mas não outros caracteres especiais (let $nome, _nome);

Mas não pode ter pontuação ou outros especiais (let @teste)

Podemos iniciar com letra maiúscula (let Nome = “Marcio”);

Ou usar camelCase (let meuPrimeiroNome = “Marcio”);

# Dúvidas ??

# EXERCÍCIOS



VAMOS Á PRÁTICA ?

# EXERCÍCIOS

Desenvolva os algoritmos para os problemas abaixo:

1) Declarar uma variável A, ler um valor para ela e escrever o valor da variável A em seguida.

2) Escreva um algoritmo que solicita ao usuário 3 valores inteiros via teclado e depois exibe os números fornecidos.

3) Efetuar a leitura de três números, some os dois primeiros e multiplique o resultado pelo terceiro → (valor1 + valor2) * valor 3

4) Efetuar a leitura de um número inteiro e apresentar o resultado do quadrado desse número.

5) Ler dois valores inteiros (variáveis A e B) e apresentar o resultado do quadrado do primeiro valor (variável A) mais o quadrado do segundo valor (variável B).

6) Ler dois valores para as variáveis A e B, e efetuar a troca dos valores de forma que a variável A passe a possuir o valor da variável B e a variável B passe a possuir o valor da variável A. Apresentar os valores trocados.

# EXERCÍCIOS



7) Desenvolva um algoritmo que receba o salário de um funcionário, calcule e mostre seu novo salário com reajuste de 15%.

8) Informe 3 números e calcule a média aritmética deles sem arredondar o valor.

9) Informe 3 números e calcule a média aritmética deles e arredonde o valor com 2 decimais.

10) Calcule quanto um cliente tem que pagar pelo seu almoço. Informe o valor por quilo e o peso que a balança mostrou. Mostre os dados conforme segue:

Preço por Kg.: R$ 99.99

Valor a Pagar: R$ 99.99