Gente,
Saúde e
Bem-estar.

RaiaDrogasil S.A.

NIVELAMENTO
LÓGICA DE PROGRAMAÇÃO

Algoritmos e Fundamentos de Linguagem de Programação

ABORDAGEM DAS AULAS

- Apresentação de conceitos (com slides)
- Apresentação de códigos de exemplo (fazendo live coding)
- Proposição de exercícios-desafios
- Avaliação (teste)

AGENDA

1. Introdução
2. Operadores
3. Variáveis
4. Funções
5. Estruturas de Decisão (If's)
6. Estruturas de Repetição (Loops)
7. Arrays

01

Introdução

INTRODUÇÃO

“Os melhores programas são escritos para que as máquinas possam executá-los rapidamente e para que os seres humanos possam entendê-los claramente. Idealmente, um programador é um ensaísta que trabalha com formas literárias e estéticas tradicionais, além de conceitos matemáticos, para comunicar a maneira como um algoritmo funciona e convencer um leitor de que os resultados estarão corretos.”

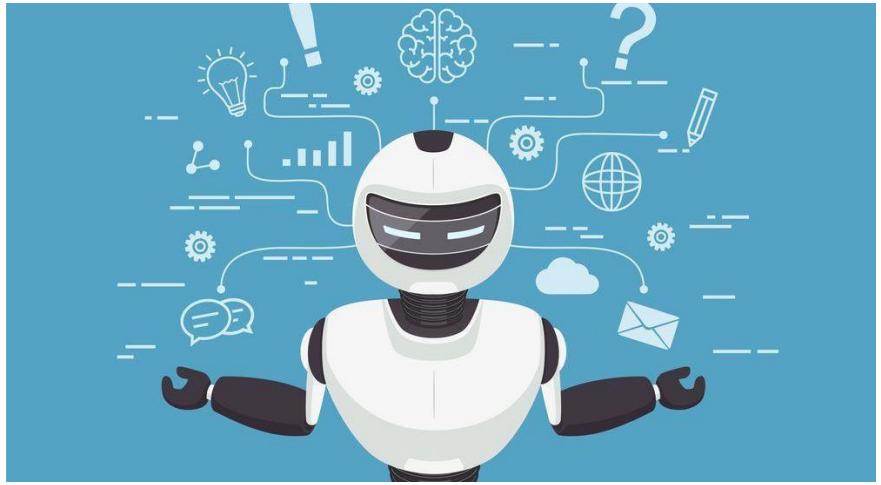
- Donald E. Knuth, in “Selected Papers on Computer Science”

INTRODUÇÃO

Algoritmo

“Na Matemática e na Ciência da Computação, um algoritmo é uma sequência finita de instruções bem definidas e implementáveis por computador, geralmente para resolver uma classe de problemas ou para executar uma computação. Os algoritmos são sempre inequívocos e são usados como especificações para executar cálculos, processamento de dados, raciocínio automatizado e outras tarefas.” [1, Wikipedia]

“Algoritmo pode ser definido como um conjunto de processos ou ações, que seguem uma sequência lógica, para executar uma tarefa.” [2, Dauricio, J.S.]



INTRODUÇÃO

Exemplo de um algoritmo para trocar pneu

1. Desparafusar a roda.
2. Suspender o carro com o macaco.
3. Retirar a roda com o pneu.
4. Colocar o estepe.
5. Abaixar o carro.
6. Parafusar a roda.

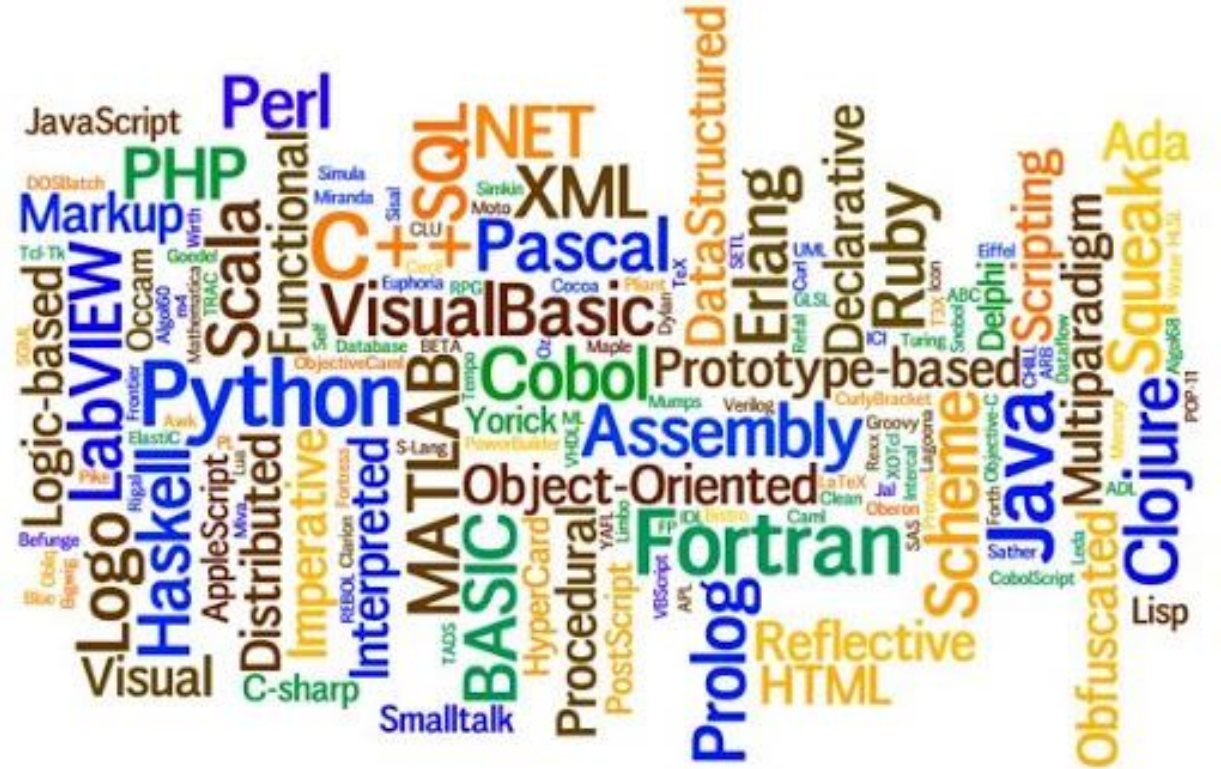
INTRODUÇÃO

Linguagem de Programação

“Uma linguagem de programação é uma linguagem formal, que compreende um conjunto de instruções que produzem vários tipos de saída. Linguagens de programação são usadas na programação de computadores para implementar algoritmos.”
[3, Wikipedia]

- São frequentemente descritas em dois componentes: sintaxe e semântica
- Podem ser definidas por uma especificação padrão; há linguagens que têm uma implementação dominante considerada como referência

[Lista das linguagens de programação]

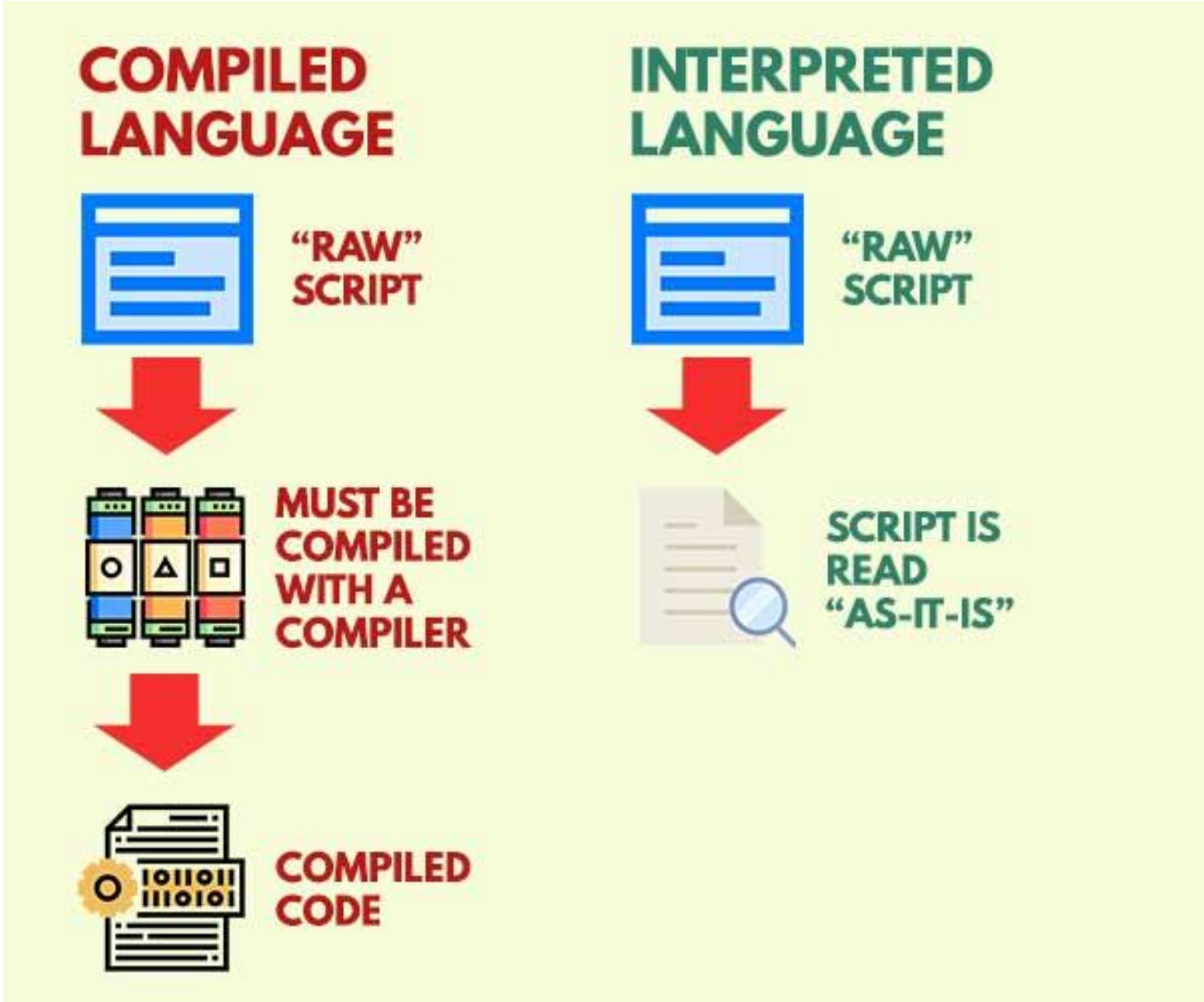


INTRODUÇÃO

Compilador vs. Interpretador [6]

Compilador: converte o código-fonte inteiro em código-objeto, um código binário que pode ser executado diretamente pela máquina após o processo de linkedição.

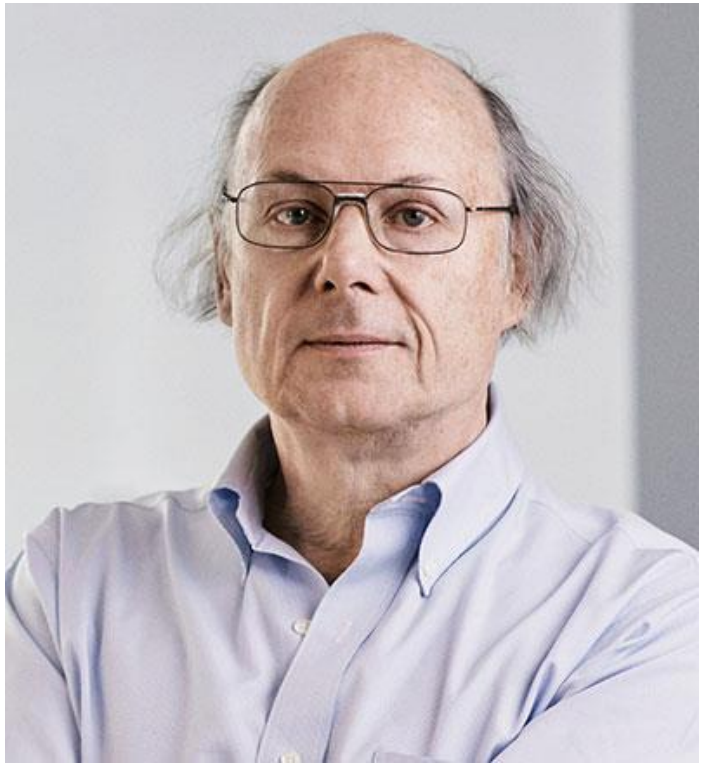
Interpretador: executa diretamente o código-fonte sem convertê-lo previamente para código-objeto ou de máquina.



INTRODUÇÃO

"Existem apenas dois tipos de linguagens: aquelas que as pessoas reclamam e aquelas que ninguém usa."

- Bjarne Stroustrup, in “The C++ Programming Language”



INTRODUÇÃO

O que será usado na aulas

HTML: “Abreviação de HyperText Markup Language (significa Linguagem de Marcação de Hipertexto) é uma linguagem de marcação utilizada na construção de páginas na Web. Documentos HTML podem ser interpretados por navegadores.” [4, Wikipedia]

JavaScript: “É uma linguagem de programação interpretada, estruturada, de script em alto nível com tipagem dinâmica fraca e multi-paradigma. Juntamente com HTML e CSS, o JavaScript é uma das três principais tecnologias da World Wide Web.” [5, Wikipedia]

Navegador: Ex.: Google Chrome

Editor de texto: Ex.: Notepad

INTRODUCAO

Exemplos

01-primeiro-teste.html
02-primeiro-teste-js.html

INTRODUÇÃO

Exercícios

01-add-alerts-scripts.md

INTRODUÇÃO

Exemplos

03-lidando-com-erros.html
04-outras-formas-comunicacao.html

02

Operadores

OPERADORES

Exemplos

05-trabalhando-com-numeros.html
06-operadores-matematicos.html

OPERADORES

Exercícios

02-testando-idades.md

03

Variáveis

VARIÁVEIS

Variável

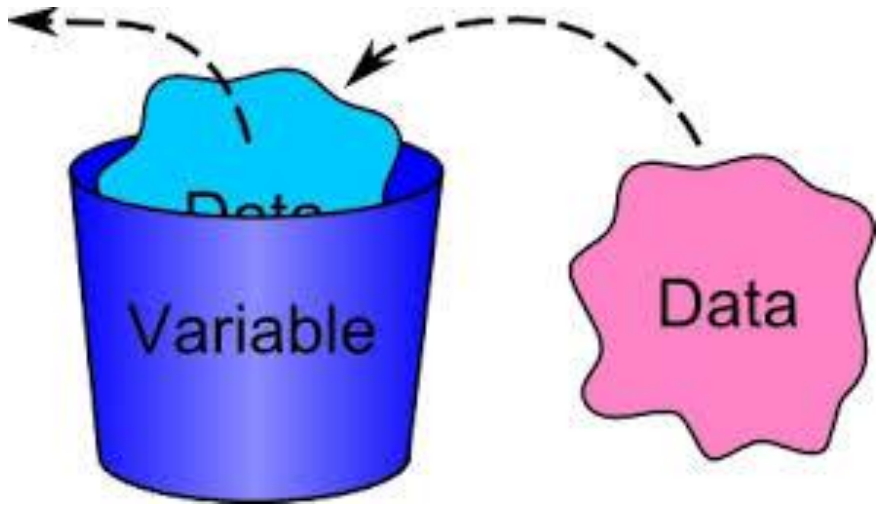
Uma variável pode guardar praticamente o que você quiser:
um número, uma string, um outro pedaço de código.

let

Palavra reservada do JavaScript, usada para criar variáveis.

Operador “=”

Usado para atribuir um valor a uma variável
Ex.: let numero = 10;



VARIÁVEIS

Exemplos

VARIÁVEIS

Exercícios

03-calcula-consumo.md

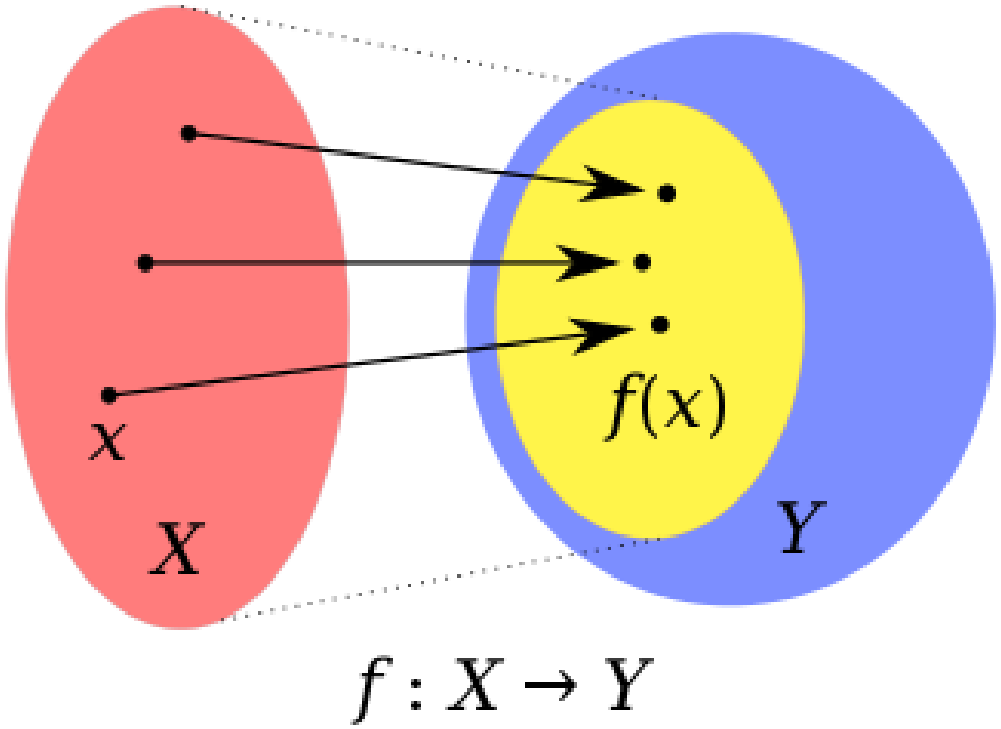
04

Funções

FUNÇÕES

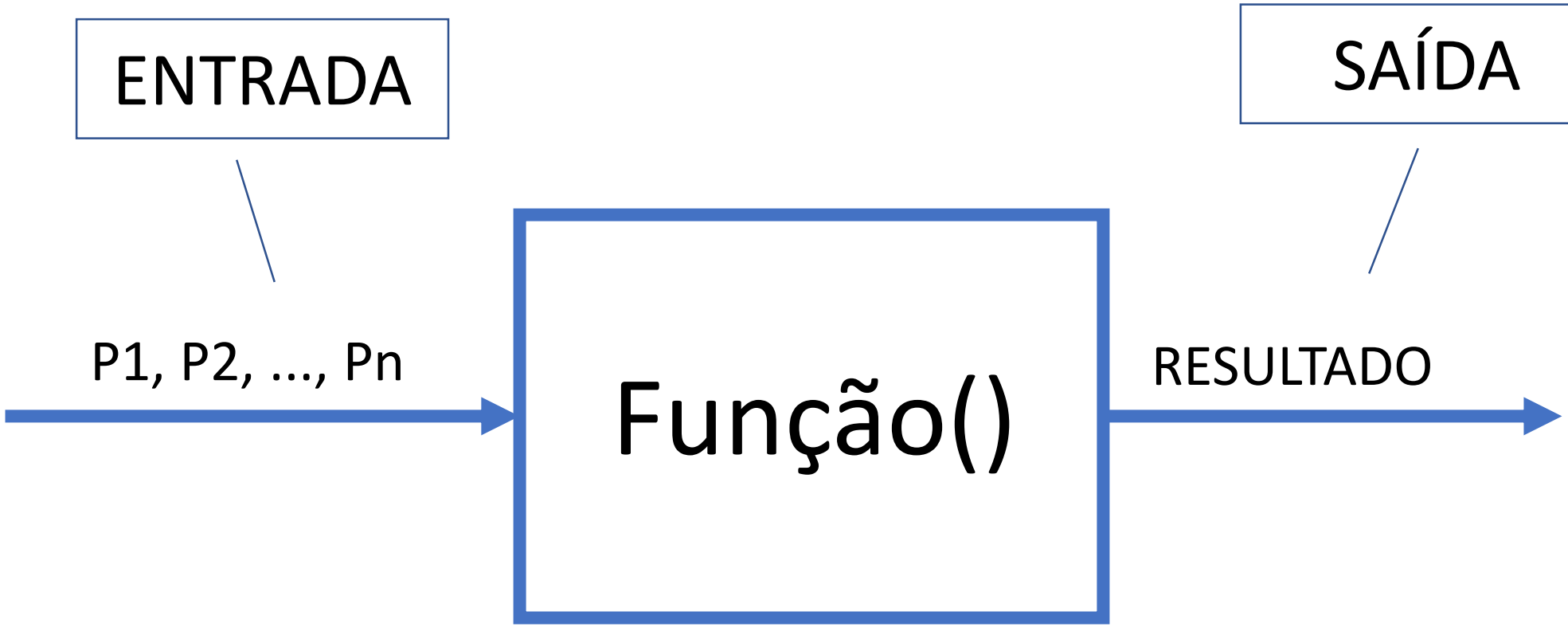
Função

É um trecho de código separado, que se pode dar um nome e chamar em outros trechos de código quando necessário. As variáveis que são passadas para as funções são frequentemente chamadas de parâmetros ou argumentos.



FUNÇÕES

Função



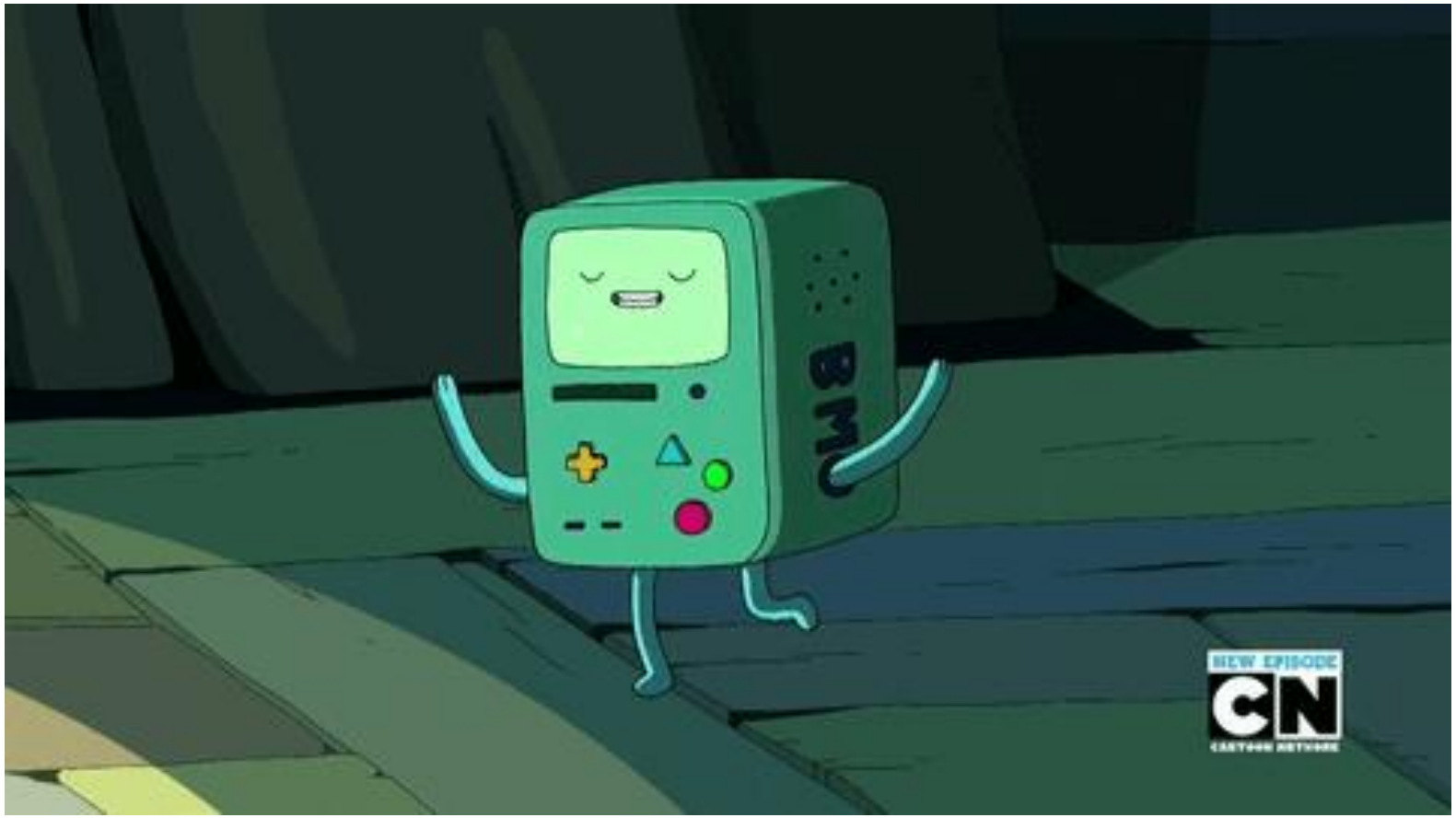
FUNÇÕES

Exemplos

08-funcoes.html
09-funcoes-parametros.html
10-mostra-idades-com-funcoes.html

FUNÇÕES

Exercícios



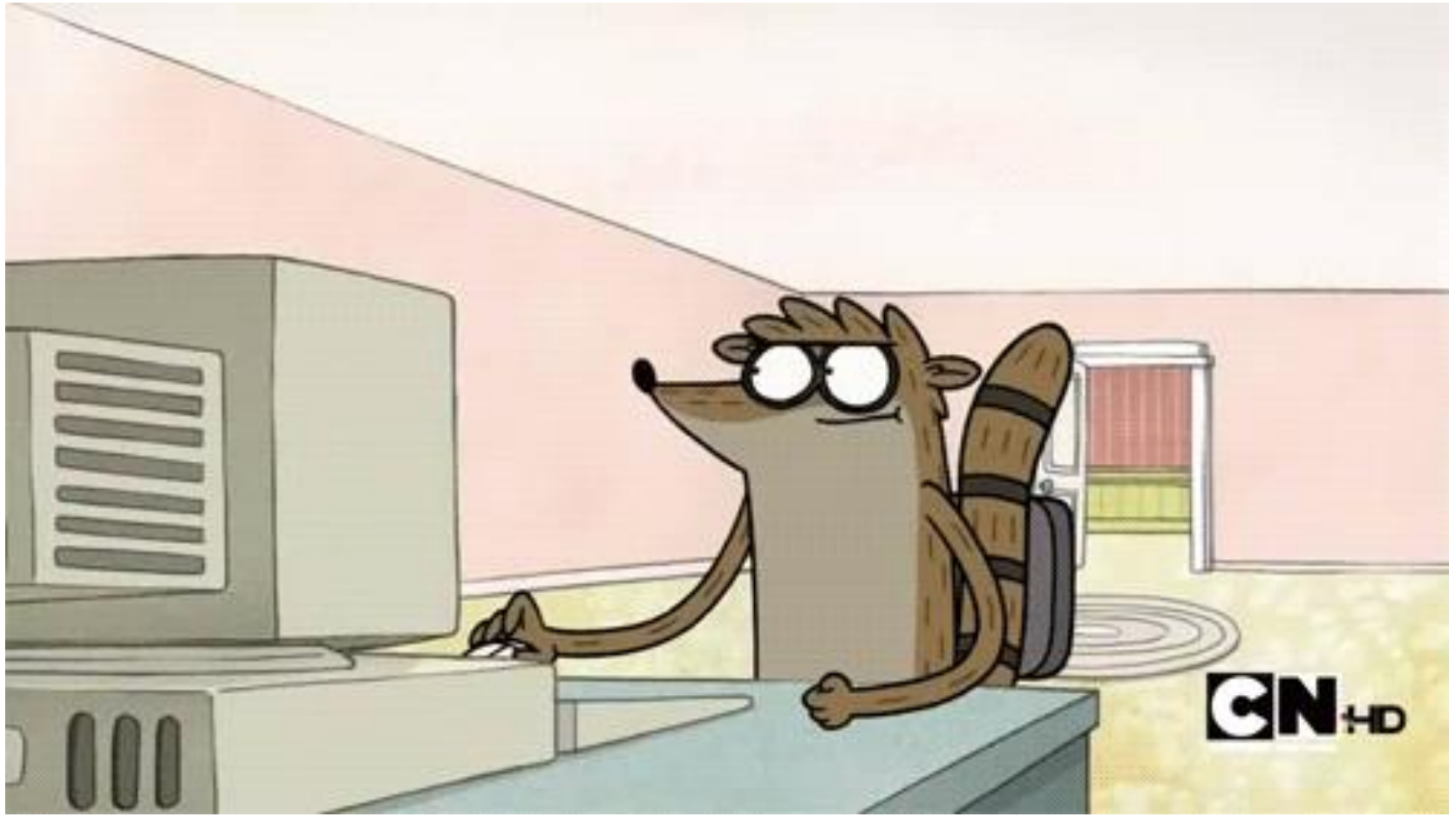
FUNÇÕES

O comando ***console.log()*** exibe uma mensagem no Console do navegador

Obs.: ***document.write()*** e ***alert()*** também são funções!

FUNÇÕES

Exemplos



11-console-log.html

FUNÇÕES

Comentários no código

```
// Esta linha é um exemplo de comentário

/*
Essa é outra forma de escrever comentários no código.

Dessa forma, pode-se usar mais linhas para escrever os
comentários.

Os comentários são ignorados pelo compilador (ou
interpretador) da linguagem
*/
```

FUNÇÕES

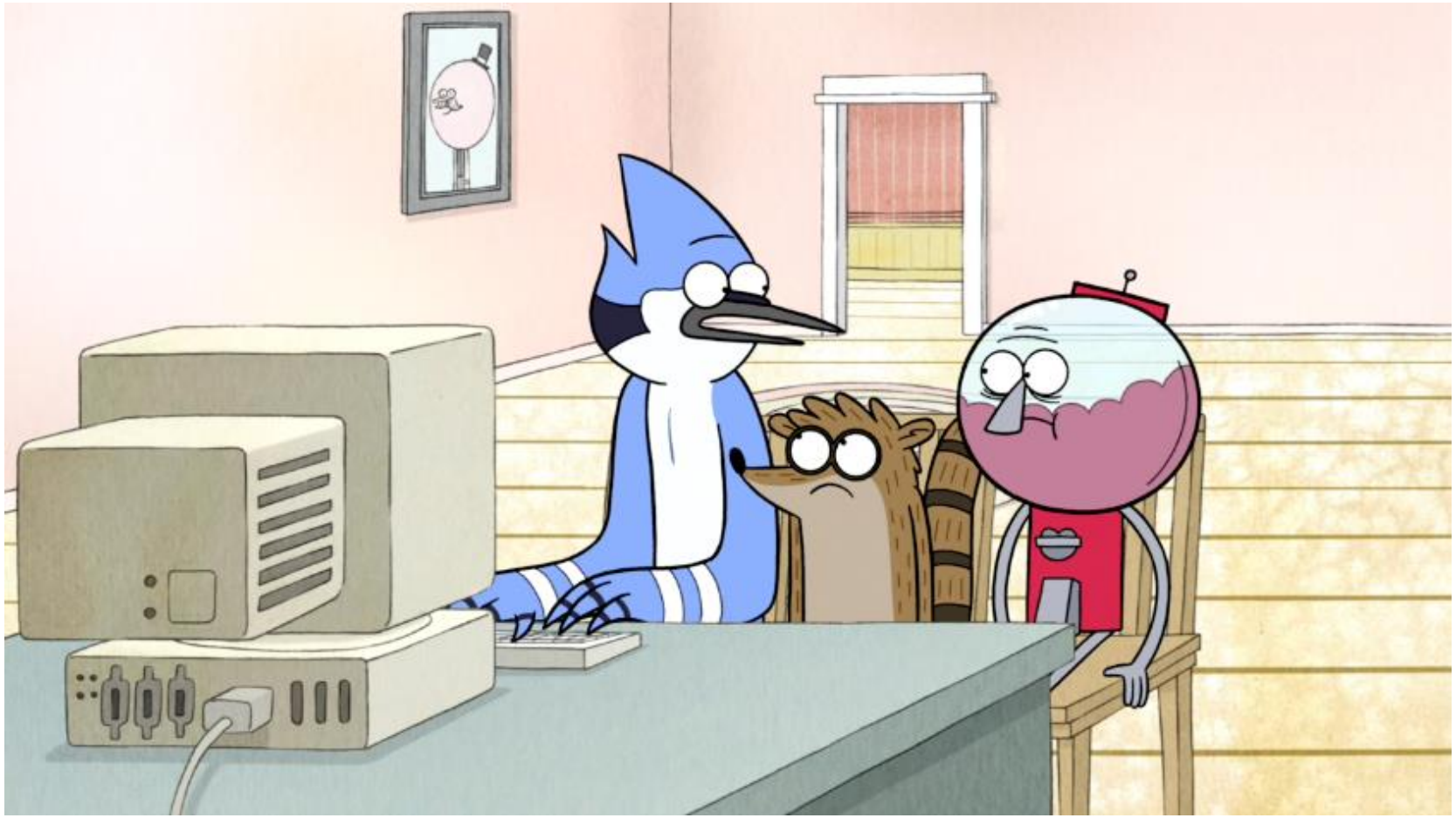
Exemplos

FUNÇÕES

Resolvendo problemas do dia a dia

FUNÇÕES

Exemplos



13-imc.html

FUNÇÕES

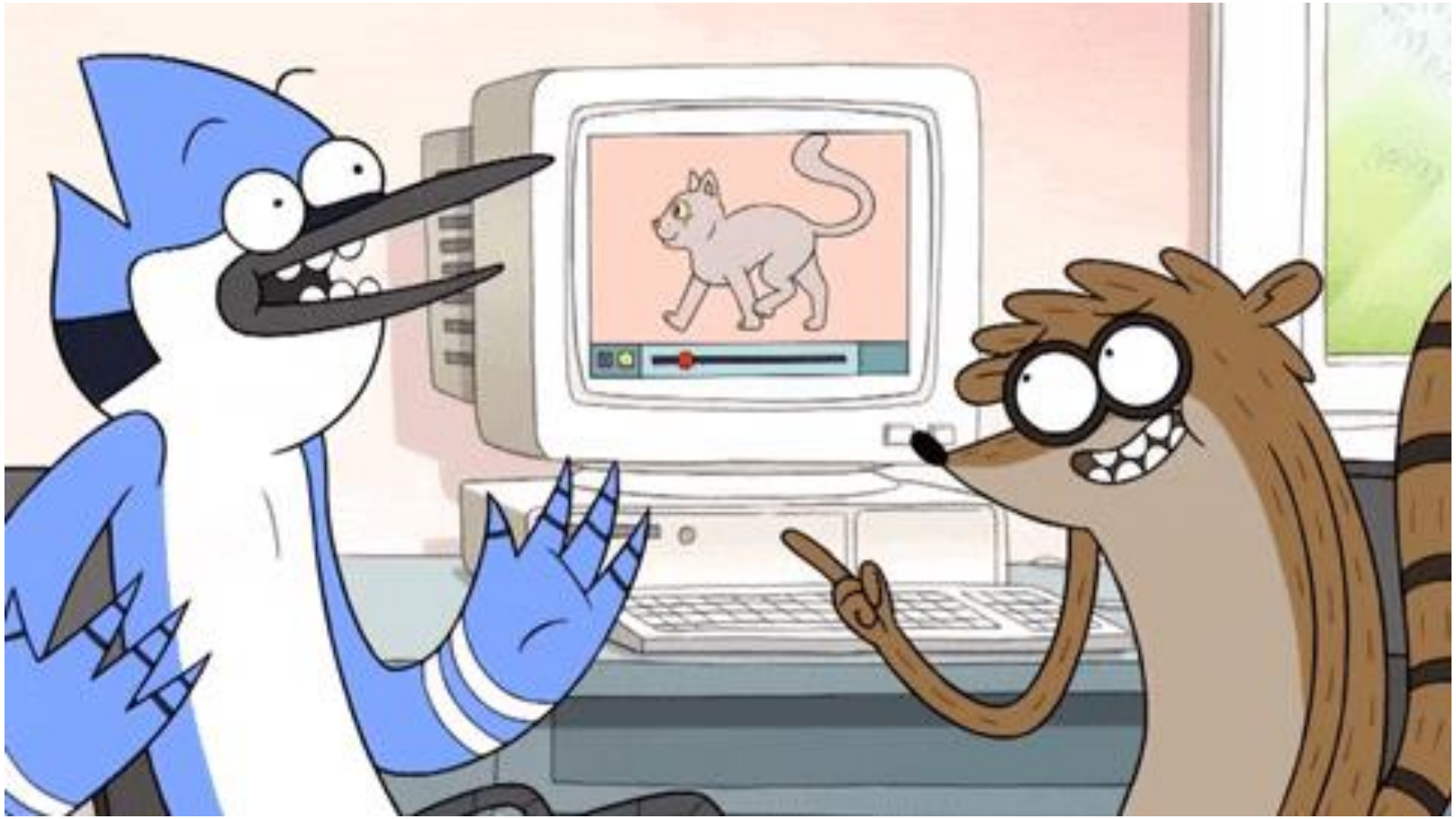
Exercícios

FUNÇÕES

Trabalhando com dados capturados do usuário

FUNÇÕES

Exemplos



14-capturando-dados.html

FUNÇÕES

Exercícios

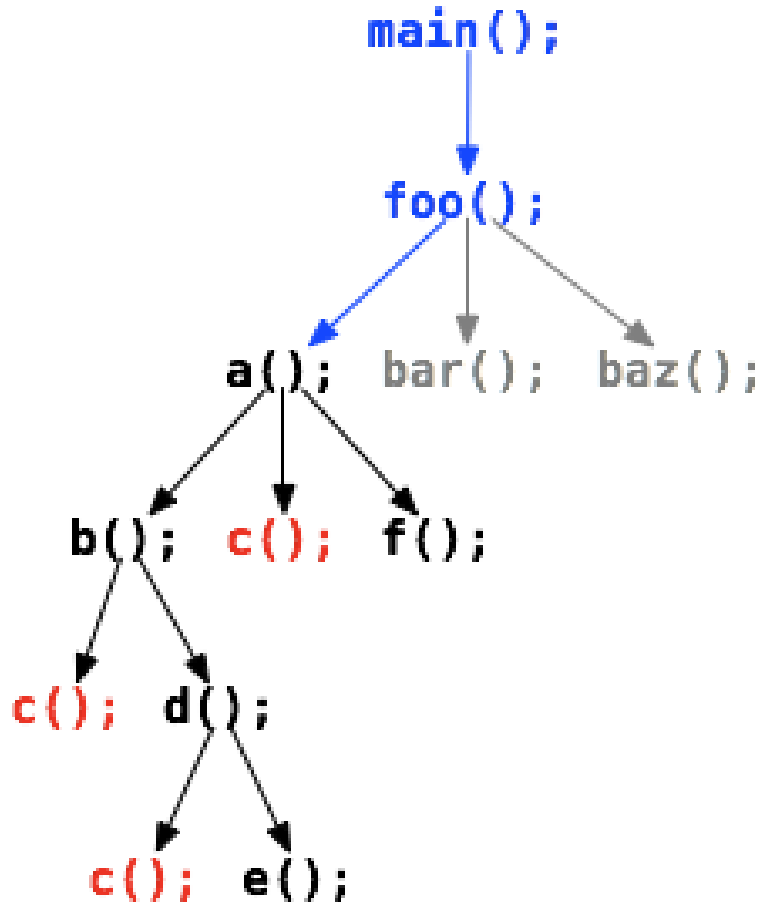
06-capturando-dados.md
07-dados-vitais.md

FUNÇÕES

Ordem das funções

```
void a() {
    b();
    c();
    f();
}

void b() {
    c();
    d();
}

void c() {
    //...
}
```

```
void d() {
    c();
    e();
}

void e() {
    //...
}

void f() {
    //...
}

//...
```



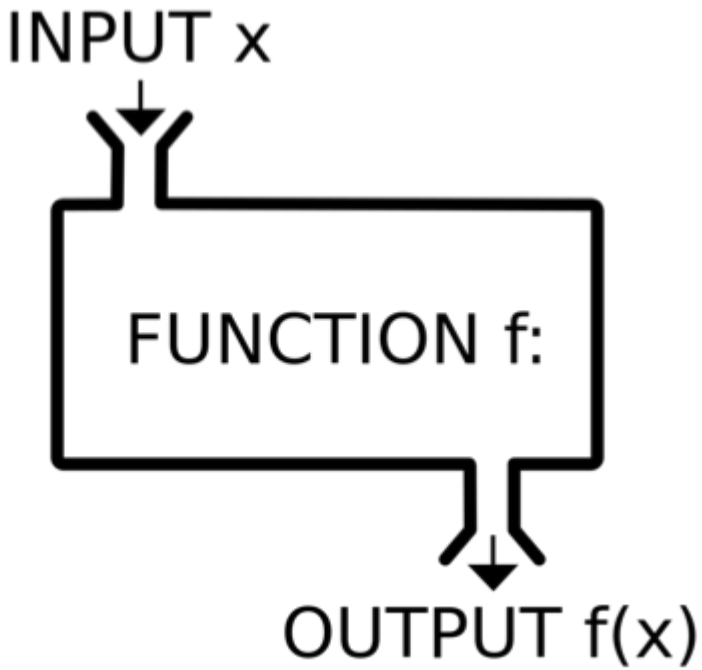
FUNÇÕES

Exemplos

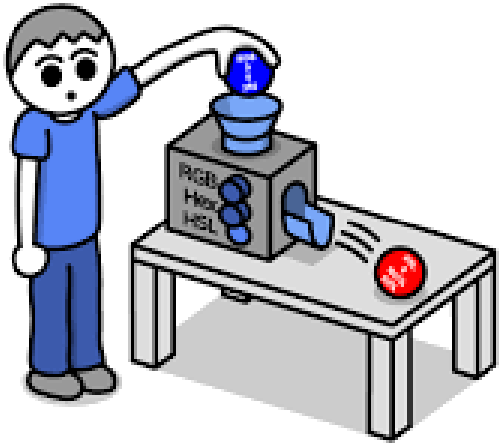
15-ordem-das-funcoes.html
15-quizz-ordem-das-funcoes.html
16-testes-no-console.html

FUNÇÕES

Outras formas de declarar funções



Convertendo valores



FUNÇÕES

Exemplos

17-formas-declaracao-funcoes.html
18-interpretador-js-e-operacoes.html

05

Estruturas de Decisão

ESTRUTURAS DE DECISÃO (If's)

Execute códigos diferentes dependendo da condição, com comando if-else if-else

Atenção: "==" é operador de comparação e "=" é operador de atribuição

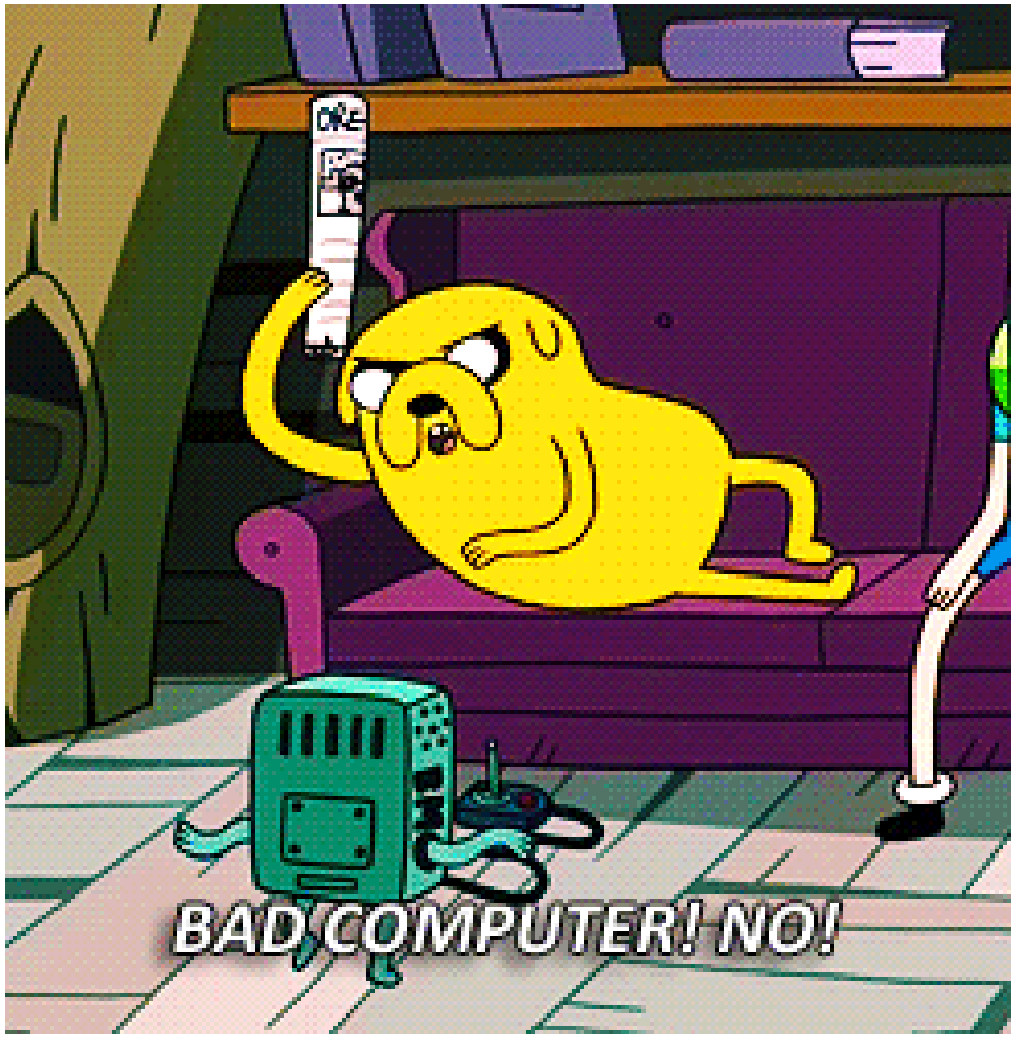
ESTRUTURAS DE DECISÃO (If's)

Exemplos

19-condicoes-ifs.html
19-condicoes-ifs-2.html

ESTRUTURAS DE DECISÃO (If's)

Exercícios



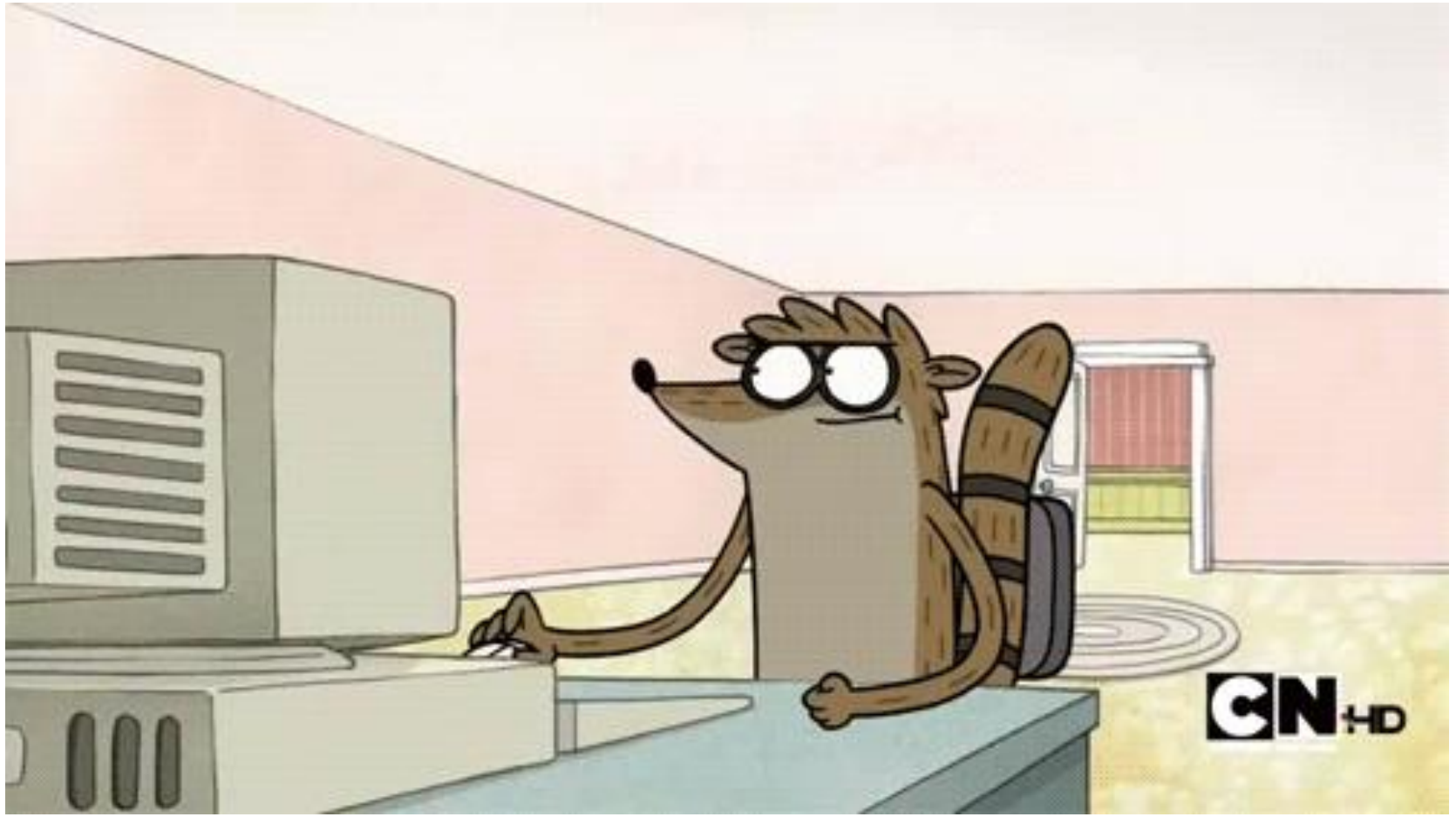
ESTRUTURAS DE DECISÃO (If's)

Outros operadores

- && (E)
- || (OU)
- >= (maior ou igual)
- <= (menor ou igual)

ESTRUTURAS DE DECISÃO (If's)

Exemplos



20-revisando-imc.html

ESTRUTURAS DE DECISÃO (If's)

Primeiro jogo!

ESTRUTURAS DE DECISÃO (If's)

Exemplos

21-primeiro-jogo.html

ESTRUTURAS DE DECISÃO (If's)

Exercícios

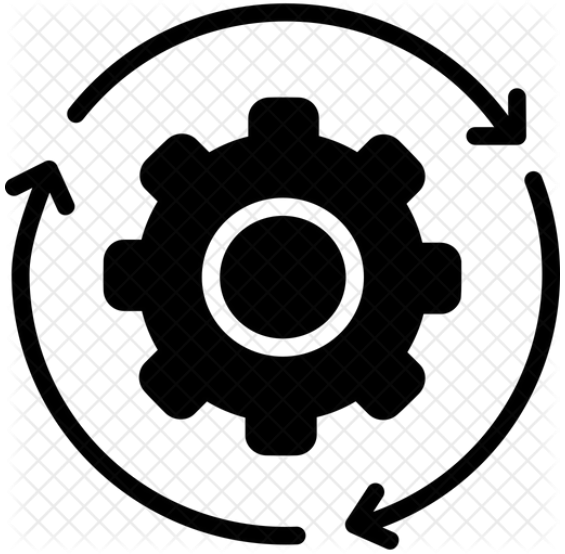
09-jogo-adivinhacao.md

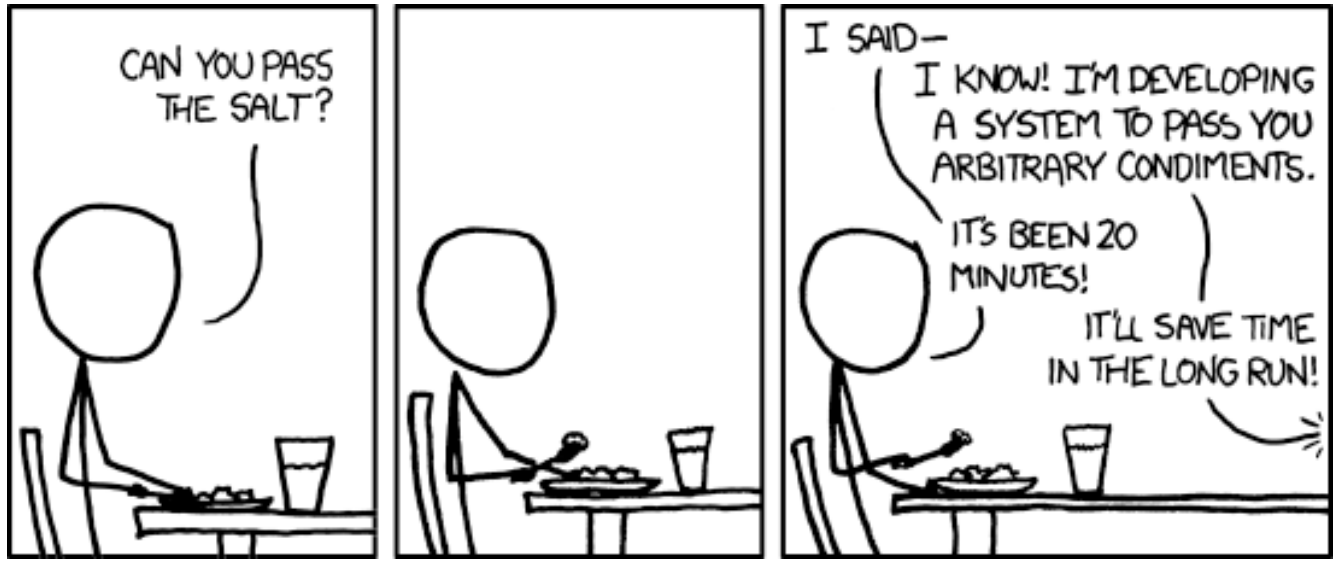
06

Estruturas de Repetição

ESTRUTURAS DE REPETIÇÃO (Loops)

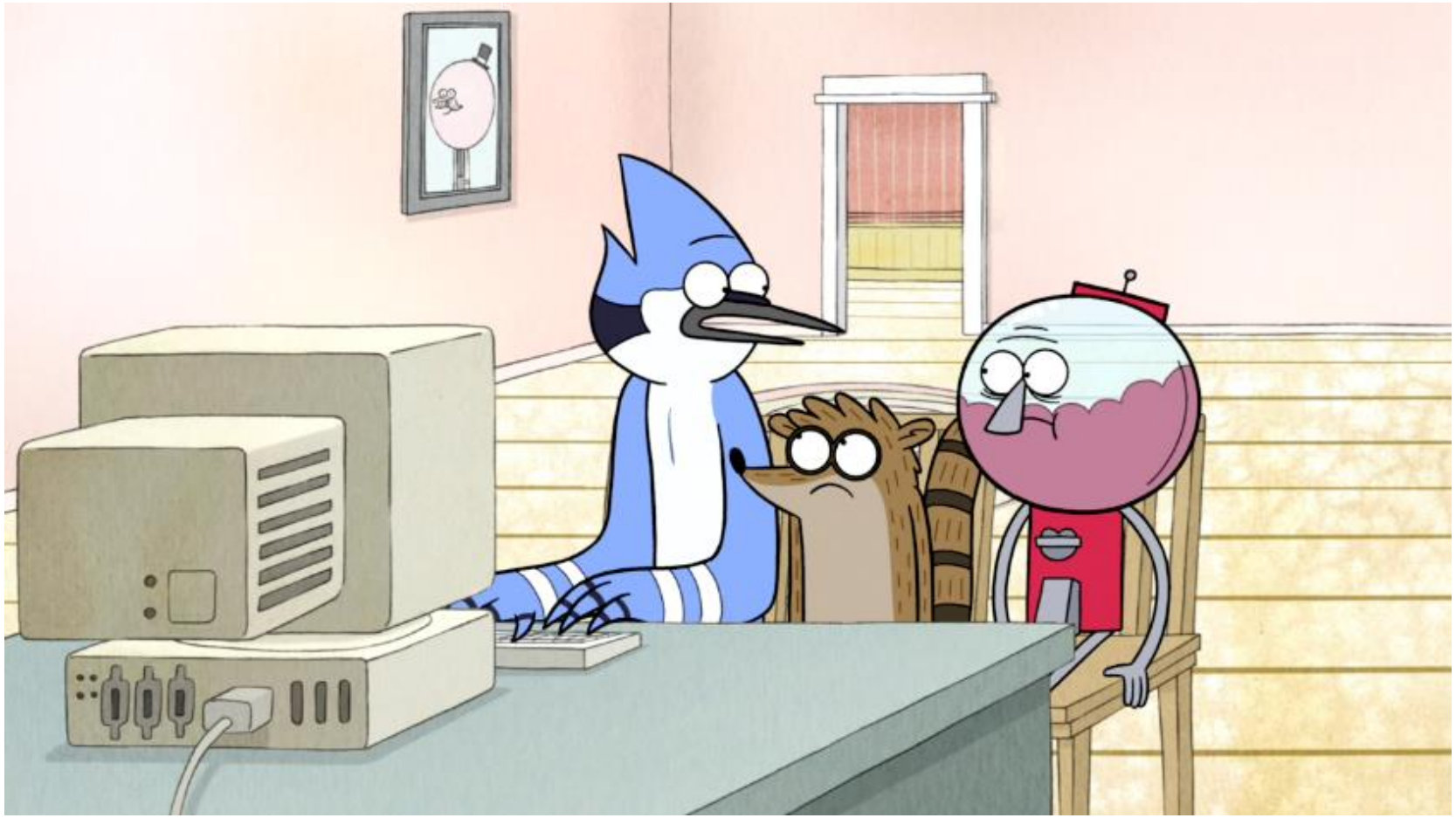
Repetir tarefas no programa e em determinadas condições





ESTRUTURAS DE REPETIÇÃO (Loops)

Exemplos



22-repeticoes.html

ESTRUTURAS DE REPETIÇÃO (Loops)

Exercícios

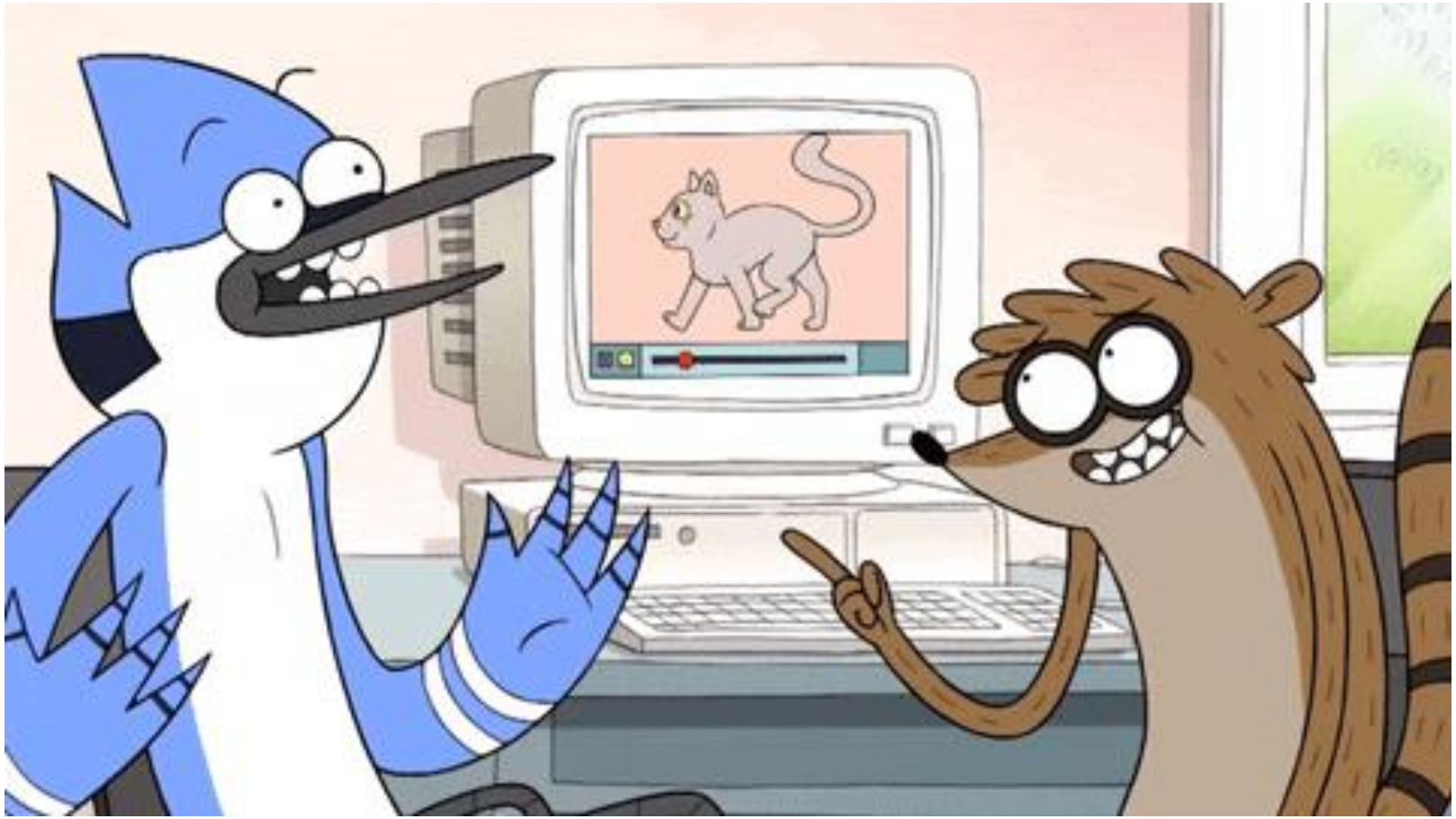
10-anos-copa-do-mundo.md
11-anos-olimpiadas.md

ESTRUTURAS DE REPETIÇÃO (Loops)

Praticando mais loops

ESTRUTURAS DE REPETIÇÃO (Loops)

Exemplos



23-tabuadas.html

ESTRUTURAS DE REPETIÇÃO (Loops)

Exercícios

12-mais-tabuadas.md

ESTRUTURAS DE REPETIÇÃO (Loops)

Uma forma diferente de fazer loop, com o comando for

As repetições que não são um loop infinito têm geralmente 3 características:

1. Um valor inicial
2. Uma condição que determina se a repetição deve ser feita ou não
3. Uma modificação no valor que causa o fim a repetição

```
<script>
    for ([inicialização]; [condição]; [modificação do valor]) {
        // código a ser repetido
    }
</script>
```

ESTRUTURAS DE REPETIÇÃO (Loops)

Exemplos

24-comando-for.html
25-tabuadas-com-for.html
26-calculo-media-com-loop.html

ESTRUTURAS DE REPETIÇÃO (Loops)

Jogo: mais chances para adivinhar o número pensado

Comando break

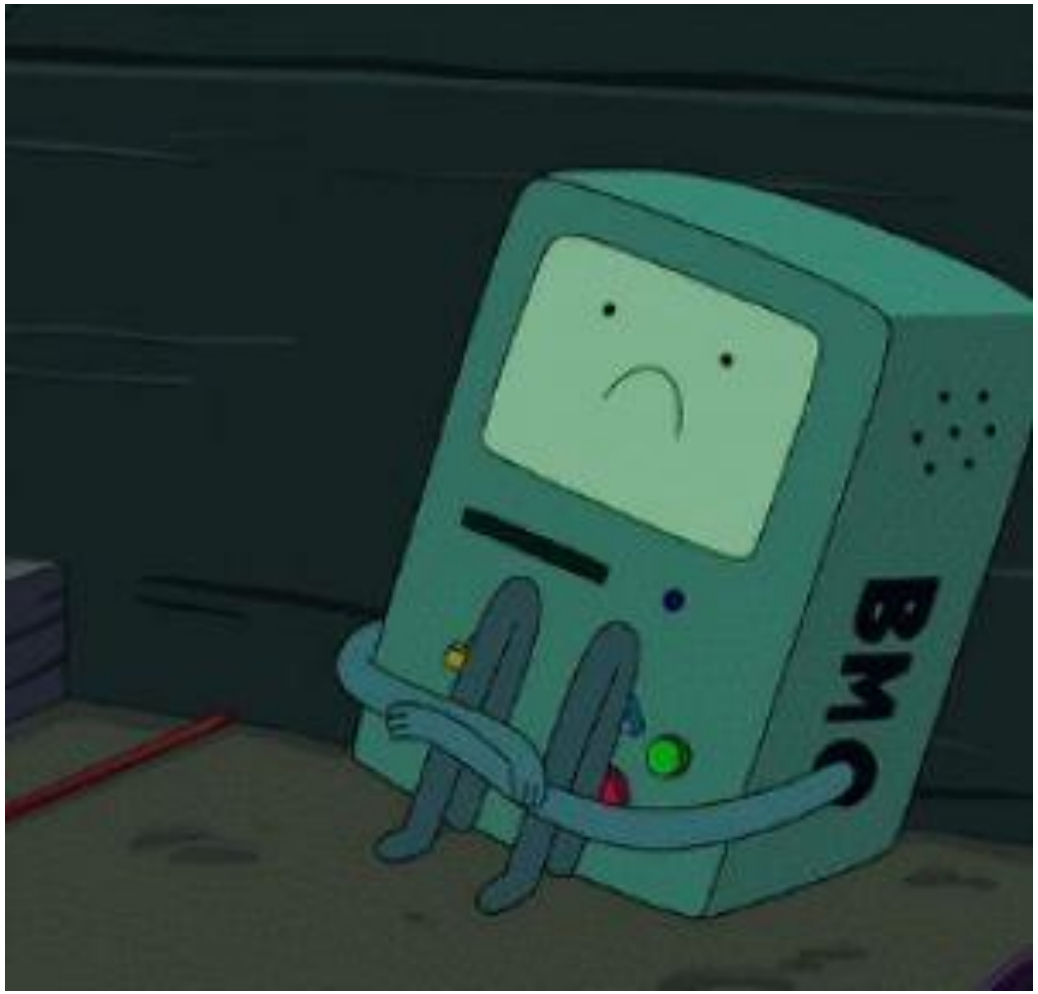
ESTRUTURAS DE REPETIÇÃO (Loops)

Exemplos

27-jogo-adivinhacao-com-tentativas.html

ESTRUTURAS DE REPETIÇÃO (Loops)

Exercícios



13-jogo-adivinhacao-v2.md

ESTRUTURAS DE REPETIÇÃO (Loops)

Loop dentro de loop, ou loop aninhados

ESTRUTURAS DE REPETIÇÃO (Loops)

Exemplos

28-loops-aninhados.html

ESTRUTURAS DE REPETIÇÃO (Loops)

Exercícios

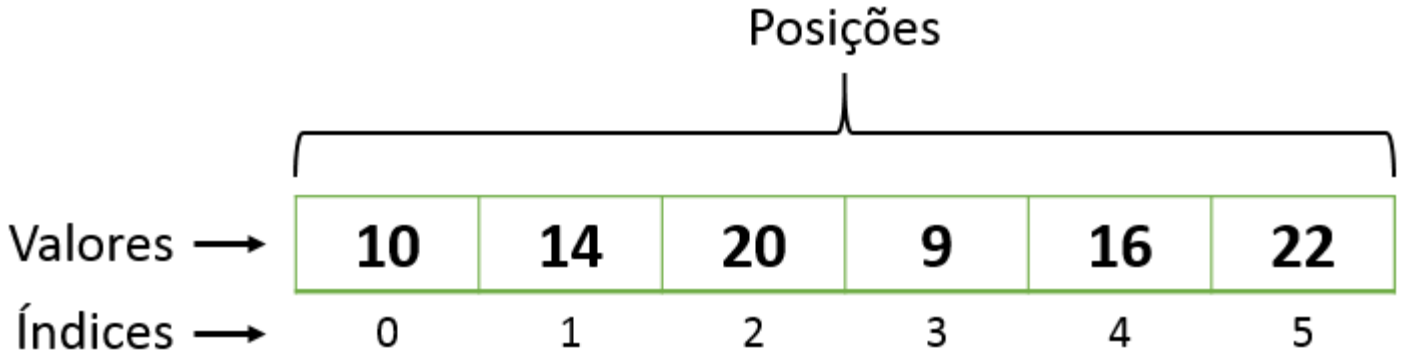
14-mais-loops-aninhados.md

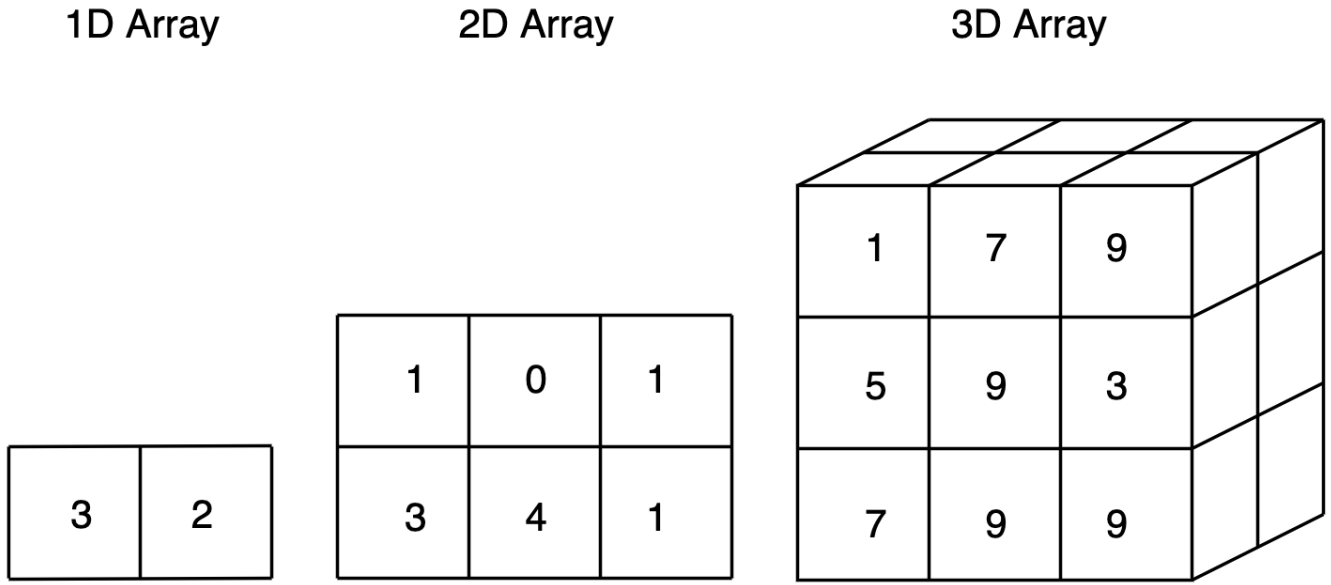
07

Arrays

ARRAYS

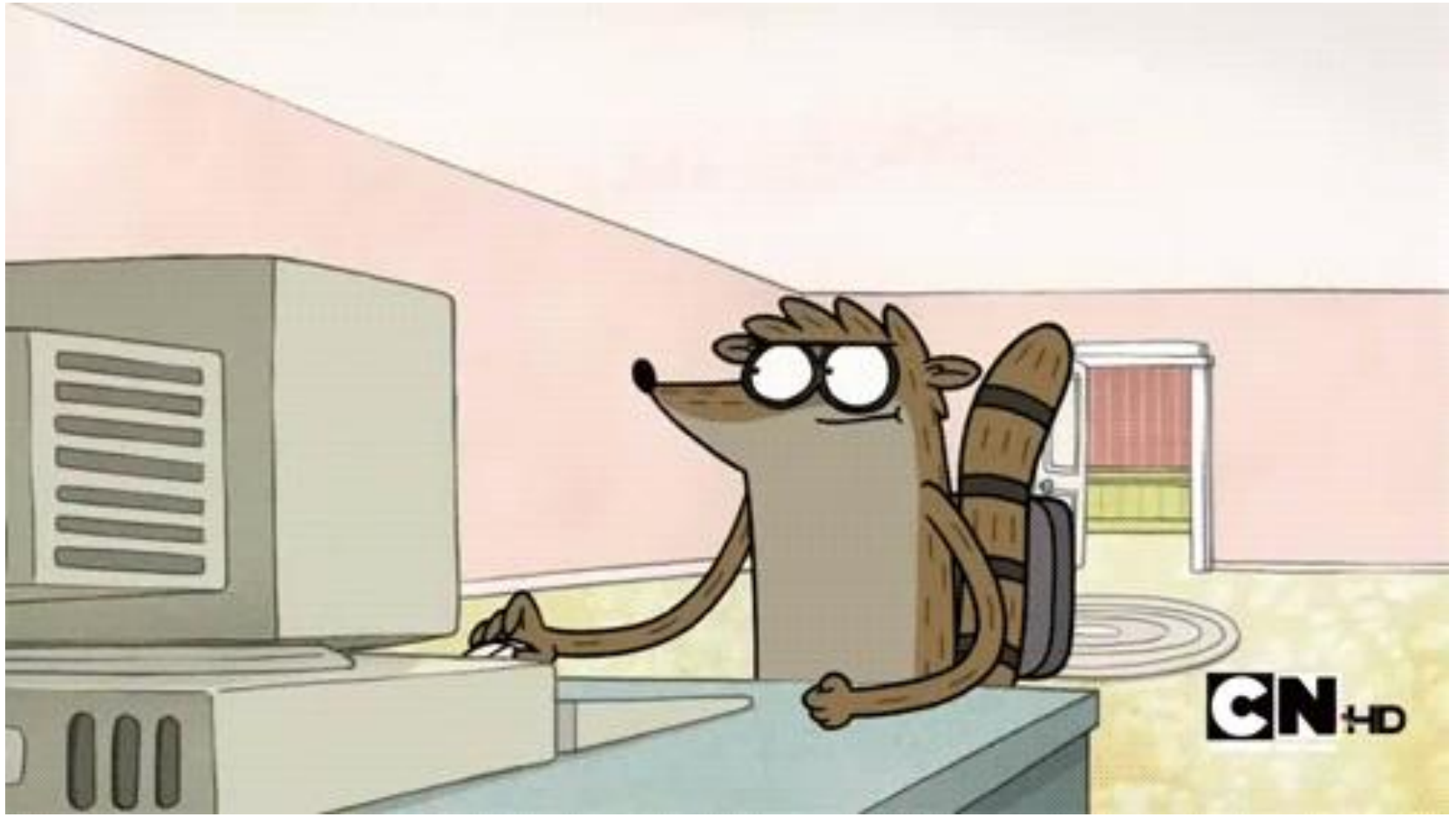
Exemplos





ARRAYS

Exemplos



29-arrays-1.html
29-arrays-2.html

ARRAYS

Exercícios

15-array-maximo-minimo.docx

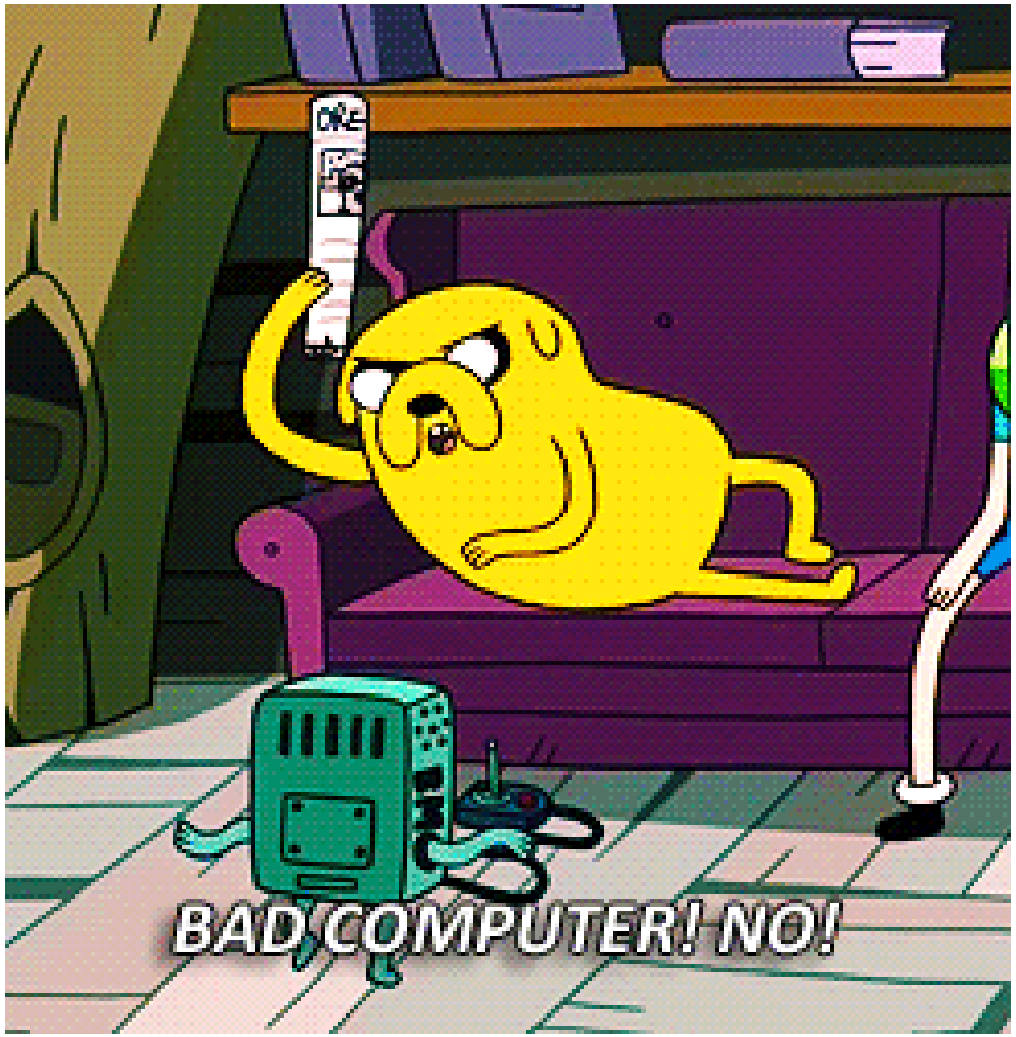
ARRAYS

Exemplos

30-array-indexof.html
31-embaralhando.html

ARRAYS

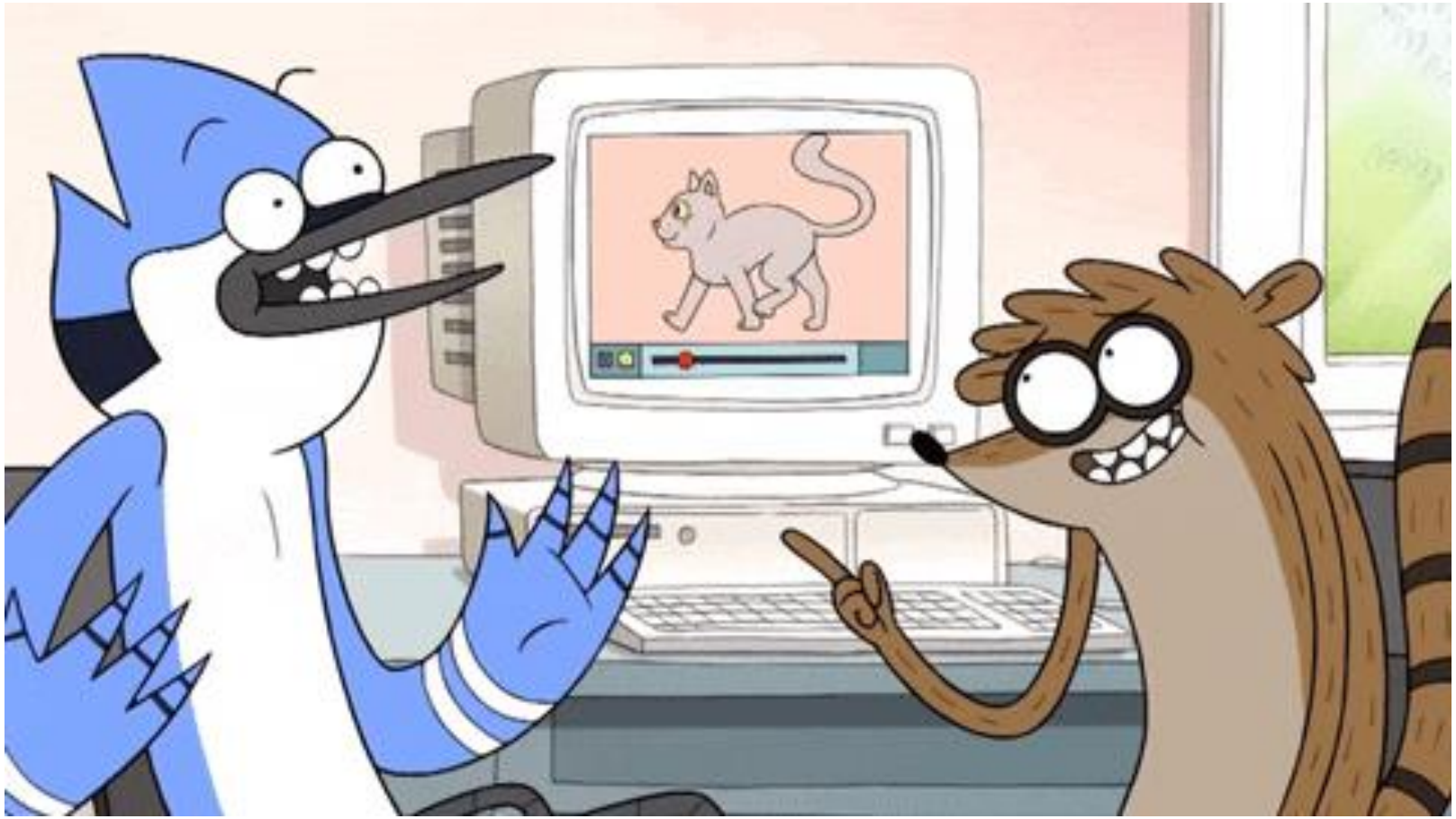
Exercícios



17-mega-sena.md

ARRAYS 2D

Exemplos



32-matriz-transposta.html

ARRAYS 2D

Exercícios

19-mundialito.docx

RECURSIVIDADE (BÔNUS)

Recursividade

"Para entender recursão, devemos primeiro entender recursão." – anônimo

É um mecanismo útil e poderoso que permite a uma função chamar a si mesma direta ou indiretamente [7].

Um algoritmo recursivo consiste em diminuir sucessivamente o problema em um problema menor ou mais simples, até que seja permitido resolvê-lo de forma direta, sem recorrer a si mesmo.

- **Vantagens**: possuem código mais claro (legível) e mais compacto do que os correspondentes iterativos.
- **Desvantagens**: consomem mais recursos (especialmente memória, devido uso intensivo da pilha) do computador, logo tendem a apresentar um desempenho inferior e são mais difíceis de serem depurados.

RECURSIVIDADE (BÔNUS)

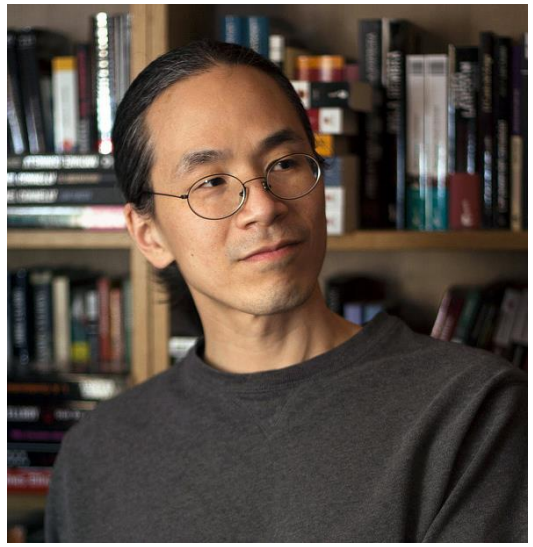
Exemplos

33-recursividade-1.html
33-recursividade-2.html
33-recursividade-3.html

MOMENTO “MÁRIO SÉRGIO CORTELLA”

“A experiência é algoritmicamente incompressível.”

- Ted Chiang, in "The Lifecycle of Software Objects“



“Mas se o pensamento corrompe a linguagem, a linguagem também pode corromper o pensamento.”

- ***George Orwell, in "1984“***

- ***Dennis Ritchie***

REFERÊNCIAS BIBLIOGRÁFICAS

Livro “Lógica de Programação - Crie seus primeiros programas usando JavaScript e HTML”, Silveira, P., Almeida, A., Ed. Casa do Código, 2017.

[1] https://en.wikipedia.org/wiki/Algorithm

[2] Livro "Algoritmos e Lógica de Programação", Dauricio, J. S., Ed. Educacional S/A, 2015.

[3] https://en.wikipedia.org/wiki/Programming_language

[4] https://en.wikipedia.org/wiki/HTML

[5] https://en.wikipedia.org/wiki/JavaScript

[6] https://www.geeksforgeeks.org/compiler-vs-interpreter-2/

[7] https://www.ic.unicamp.br/~ripolito/peds/mc102z/material/Recursividade.PDF

FIM

Obrigado!

☺