Algoritmos e Estrutura de um Programa

- Algoritmo: é uma sequência de raciocínios, instruções ou operações para alcançar um objetivo.
- Jogo para exercitar lógica de programação: https://blockly.games/maze
- Compute IT e Silent Teacher

Referências:

- Algoritmo: o que é, como funciona e quais são os principais exemplos

O que é JS?

- Linguagem de programação interpretada criada em 1995.
- Tipagem dinâmica fraca.
- Multi Paradigma (procedural, POO, Funcional).
- Evolução constante (ES6+): Praticamente uma vez por ano é lançada uma revisão/atualização da linguagem.

Referências:

- O que é JavaScript? Conheça o funcionamento e vantagens - TOTVS
- Introdução - JavaScript | MDN (mozilla.org)
- O que é JavaScript? Como usar o JavaScript? (celke.com.br)

Atribuição, Variável e Constante

- Declaração de variáveis e constantes: cria um espaço na memória onde pode ser armazenado valores para reuso posteriormente.
    - Variável: valor pode ser alterado no futuro,
    - Constante: Após declaração (e atribuição) o valor não pode ser alterado.
- [EXEMPLO]:

```
let x
console.log(x)
x = 10
console.log(x)
```

- [EXEMPLO]:

```
const y
console.log(y)
```

- [EXEMPLO]:

```
const y = 10
console.log(y)
```

- [EXEMPLO]:

```
const y = 10
let x
```

```
x = y + 12
console.log(x)
x = x + 15
console.log(x)
```

Referências:

- JavaScript: Variáveis e constantes (devmedia.com.br) - (tem uns exemplos muito bom)
- Entenda a diferença entre var, let e const no JavaScript | Alura Cursos Online
-

Tipos Primitivos

- Number
  - Dentro do JS o tipo Number engloba inteiros, decimais, negativos e positivos.
  - [EXEMPLO]:

```
console.log(42)
console.log(1.99)
console.log(-9)
```

- String
  - Valores alfanuméricos entre aspas simples ou duplas.
  - [EXEMPLO]:

```
console.log('Hello World')
console.log('')
console.log(' ')
console.log('\n')
console.log('42')
```

- Boolean
  - Valores que representam verdadeiro ou falso (true e false)
  - [EXEMPLO]:

```
console.log(true)
console.log(false)
```

- null
  - Valor que representa nulidade de valor
  - [EXEMPLO]:

```
console.log(null)
```

- undefined
  - Valor que representa um valor ainda não definido.
  - [EXEMPLO]:

```
console.log(undefined)
```

Referências:

- Primitivo - Glossário | MDN (mozilla.org)

Concatenar Strings

- Concatenar Strings (+)
  - [EXEMPLO]:

```
console.log('Otter' + 'wise')
console.log('Hello' + ' ' + 'World')
```

- O que acontece no JS se tentarmos concatenar uma String com um Number ?
  - [EXEMPLO]:

```
console.log('Hello' + 12)
console.log('42' + 12)
console.log(12 + '42')
```

Referencia:

- Javascript - O operador + | Da2k Blog

Expressões Aritméticas

- Principais expressões:
  - Soma (+)
    - [EXEMPLO]:

```
console.log(42 + 12)
console.log(-10 + 5 + 12  + -5)
```

  - Subtração (-)
    - [EXEMPLO]:

```
console.log(42 - 12)
console.log(-10 - 5 - 12  - -5)
```

  - Multiplicação (*)
    - [EXEMPLO]:

```
console.log(42 * 12)
console.log(5 * 10 + 5)
```

    - Precedência e parênteses
      - [EXEMPLO]:

```
console.log(5 * 10 + 5)
console.log(5 * (10 + 5))
```

    - [EXEMPLO]:

```
console.log(-10 * 5 * -5)
```

  - Divisão (/)
    - [EXEMPLO]:

```
console.log(42 / 12)
```

```
console.log(42 * 10 / 2) // trocar precedencia
```

- Potência (**)
  - [EXEMPLO]:

```
console.log(3 ** 3)
```

- Resto (%)
  - [EXEMPLO]:

```
console.log(42 % 12)
```

Referências:

- Expressões e operadores - JavaScript | MDN (mozilla.org)
- JavaScript Aritmética (w3bai.com)

Checkpoint de Exercícios 1

- Crie variáveis para armazenar algumas informações de cadastro de um estudante, são elas: nome, sobrenome, dia do nascimento, mês do nascimento, ano do nascimento, primeira nota e segunda nota. A partir dessas informações, você deve mostrar um relatório na tela da seguinte forma: Nome Completo, Idade, Data de Nascimento e Média:

  Exemplo de Saída:
  Nome Completo: João Pedro Bretanha
  Idade: 29
  Data de Nascimento: 11/04/1992
  Média: 8

- Crie um programa que some os números 5, 10 e 15. Salve o resultado em uma variável e imprima no console.

Operadores de Comparação

- Igual (==) e estritamente igual (===)
  - [EXEMPLO]

```
let x = 12
console.log(x === 12)
console.log(x == '12')
console.log(x === '12')
```

- Diferente (!=) e estritamente diferente (!==)
  - [EXEMPLO]

```
let x = 12
console.log(x !== 12)
console.log(x != '12')
console.log(x !== '12')
```

- Menor (<) e menor e igual (<=)
    - [EXEMPLO]

```
let x = 12
console.log(x < 12)
console.log(x <= 12)
```

- Maior (>) e maior e igual (>=)
    - [EXEMPLO]

```
let x = 12
console.log(x > 12)
console.log(x >= 12)
```

Referências:

- Estruturas condicionais Javascript - Todo Espaço Online (todoespacoonline.com)
- Tomando decisões no seu código — condicionais - Aprendendo desenvolvimento web | MDN (mozilla.org)

Condicionais

- Instruções condicionais no JS:
    - IF
        - [EXEMPLO]:

```
const x = 10
if (x < 0) {
    console.log('valor negativo')
}
```

    - IF com ELSE
        - [EXEMPLO]:

```
const x = 10
if (x < 0) {
  console.log('valor negativo')
} else {
  console.log('valor positivo')
}
```

    - IF com ELSE IF
        - [EXEMPLO]

```
if (x < 0) {
  console.log('valor negativo')
} else if (x > 0) {
  console.log('valor positivo')
} else {
  console.log('valor zero')
```

```
}
// ------------------------------------------------
if (op === '1') {
  console.log('Escolhida opção 1')
} else if (op === '2') {
  console.log('Escolhida opção 2')
} else if (op === '3') {
  console.log('Escolhida opção 3')
} else {
  console.log('opção inválida')
}
```

- SWITCH CASE
  - [EXEMPLO]:

```
const op = '1'
switch (op) {
  case '1':
    console.log('Escolhida opção 1')
    break
  case '2':
    console.log('Escolhida opção 2')
    break
  case '3':
    console.log('Escolhida opção 3')
    break
  default:
    console.log('opção inválida')
    break
}
```

Expressões Lógicas

- And (&&)
  - As duas condições devem ser verdadeiras para que a expressão inteira seja verdadeira.
  - [EXEMPLO]:
    - true && true é avaliado como verdadeiro
    - true && false é avaliado como falso
    - false && true é avaliado como falso

        - false && false é avaliado como falso
- Or (||)
    - Apenas uma das condições precisa ser verdadeira para que a expressão inteira seja verdadeira.
    - [EXEMPLO]:
        - true && true é avaliado como verdadeiro
        - true && false é avaliado como verdadeiro
        - false && true é avaliado como verdadeiro
        - false && false é avaliado como falso
- Not (!)
    - [EXEMPLO]:
        - !true é avaliado para falso
        - !false é avaliado para true

Referências

- Operadores - JavaScript | MDN (mozilla.org) (final da página)
- Operadores Lógicos AND (&&), OR (||) e NOT (!) em JavaScript - JavaScript Progressivo - (tem tabela verdade)

Checkpoint de Exercícios 2

1. Faça um programa que diga se um número é positivo ou negativo. Mostre “positivo” ou “negativo” de acordo com o número testado.

2. Crie um programa que lê dois valores, x e y, e diz se algum desses valores é positivo.

3. Elabore um algoritmo que calcule o que deve ser pago por um produto, considerando o preço normal de etiqueta e a escolha da condição de pagamento. Utilize os códigos da tabela a seguir para ler qual a condição de pagamento escolhida e efetuar o cálculo adequado, imprimindo o valor final no console.
   Código Condição de pagamento :
   1 - À vista em dinheiro ou cheque, recebe 10% de desconto
   2 - À vista no cartão de crédito, recebe 15% de desconto
   3 - Em duas vezes, preço normal de etiqueta sem juros
   4 - Em três vezes, preço normal de etiqueta mais juros de 10%

Função Regular

- Pode-se isolar trechos de códigos em funções para reuso posteriormente.
- Aqui é bem importante trazer analogias para cada tipo de função:
    - Sem parâmetro e sem retorno;
    - Sem parâmetro e com retorno;
    - Com parâmetro e sem retorno;
    - Com parâmetro e com retorno;
- Funções podem ter parâmetros, que são os valores de entrada da função.
- Funções podem ter um retorno, que é o valor de saída da função.

- Funções são definidas e posteriormente são chamadas.
- Na definição da função deve ser definido seu nome, seus parâmetros e o bloco de código que será executado quando ela for chamada.
- Chamada da função é quando o bloco de código será de fato executado recebendo seus argumentos e retornando seu resultado.
    - [EXEMPLO]

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

console.log(soma(5, 3))
const resultado = soma(-10, 50)
console.log(resultado)
```

- Parâmetros e Argumentos.
- Funções podem, ou não, ter um retorno explícito.
    - [EXEMPLO]

```
function print(value) {
  console.log(value)
}
print('Hello World')
const retorno = print('Hello World')
console.log(retorno)
```

- Funções podem, ou não, ter parâmetros.
    - [EXEMPLO]

```
function getHello() {
  return 'Hello World'
}
const retorno = getHello()
console.log(retorno)
```

- Qualquer valor válido no JS pode ser passado como argumento na chamada de uma função:
    - String, Number, Boolean, null, undefined
    - Constantes e Variáveis
    - Expressões
- Da mesma forma, qualquer valor válido no JS pode ser retornado em uma função.
    - [EXEMPLO]

```
function selectOp(op) {
  switch (op) {
    case '1':
      return 'Escolhida opção 1'
    case '2':
```

```
        return 'Escolhida opção 2'
      case '3':
        return 'Escolhida opção 3'
      default:
        return 'opção inválida'
  }
  const op = '1'
  console.log(selectOp(op))
```

Referências:

- Funções - JavaScript | MDN (mozilla.org)
- Funções – JavaScript | Dev Content (Bem explicativo, muito completo, tem um trecho sobre rest operator, fala sobre escopo tbm)

Escopo (10 min)

- Espaço onde as variáveis, constantes e funções são acessíveis.
- São delimitados por chaves.
- O próprio arquivo JS tem seu escopo global.
- Blocos de condicionais e funções definem escopo
- [EXEMPLO]

```
let num = 10
if (num < 10) {
  num = 20
  console.log(num)
}
console.log(num)
// ------------------------------------------------------------
let num = 10
if (num < 10) {
  let num = 20
  console.log(num)
}
console.log(num)
```

Referências

- JavaScript Escopo (w3bai.com) (tem o tente você mesmo, bem bom)
- Entendendo o uso de escopo no JavaScript | by Yure Pereira | Weyes | Medium

Exercícios Finais (17 min)

1. Faça uma função que recebe um valor inteiro e verifica se o valor é par. A função deve retornar um valor booleano se for par.
2. Faça uma função que recebe a idade de um nadador e retorna a categoria desse nadador de acordo com a tabela abaixo:
   Idade - Categoria
   5 a 7 anos - Infantil A
   8 a 10 anos - Infantil B
   11 - 13 anos - Juvenil A
   14 - 17 anos - Juvenil B
   Maiores de 18 anos (inclusive) - Adulto