Estruturas de Repetição

- A instrução *for* é a estrutura mais básica que temos no JS.
- O for cria um laço de repetição e na sua construção vamos colocar o **inicializador**, a **condição**, a **expressão** e o **bloco de código**.
    - [EXEMPLO] Algoritmo que imprime n vezes “Hello World”

```
let n = 3
for (let i = 0; i < n; i++) {
  console.log('Hello World')
}
```

Referências

- Loop For – Estruturas De Repetição Em Javascript - Celso Kitamura

Exercícios

1. Imprima no console 15 vezes a frase ‘Formação Otterwise’.

Array

- Arrays são listas de elementos.
- Pode-se construir listas com elementos de qualquer valor válido no JS, como string, number, boolean, null, undefined e função.
- Para criarmos um array, envolvemos os elementos entre chaves e os separamos por vírgula.
    - [EXEMPLO]:

```
let array = [10, 20, 30]
let array2 = []
const array3 = ['agua', 'maça', 'pao']
```

    - [EXEMPLO] Alterar o valor do array
- Podemos alterar os valores dos elementos do array também. Para isso precisamos primeiro acessar esse elemento.
- Elementos de um array são acessíveis a partir de seu índice (posição no array). A primeira posição do array é sempre 0.
    - [EXEMPLO]

```
console.log(array[0])
console.log(array[0], array[1], array[2])
console.log(array[3])
```

- Vamos tratar esse acesso como uma “variável”, então podemos alterar esse elemento atribuindo um novo valor a esse endereço.

    - [EXEMPLO]:

```
array[0] = 50
array[1] = array[1] * 10
```

- Array é uma **classe** no JS, assim como Number, String, Boolean, etc… Sempre que criamos um array, estamos criando uma instância dessa classe Array.
- Toda instância da classe array tem um atributo chamado **length** que nos indica o número de elementos no array (tamanho do array).
    - [EXEMPLO]

```
console.log(array.length)
```

Referências

- Array - JavaScript | MDN (mozilla.org)
- JavaScript Arrays (w3schools.com)

Estrutura de repetição + Array

- Estruturas de repetição são instruções que permitem executarmos um bloco de código mais de uma vez. permitem inclusive que as informações desse bloco mudem entre as execuções.
    - [EXEMPLO]

```
console.log(array[0])
console.log(array[1])
console.log(array[2])
```

- E se tivermos 100 posições no array?
    - [EXEMPLO]

```
let array = [10, 20, 30]
for (let i = 0; i < array.length; i++) {
  console.log(array[i])
}
```

Referências

- Estruturas de Repetição no Javascript - The Relicans

Exercícios

1. Crie um algoritmo que imprime todos os valores de um array qualquer, menos o ultimo elemento.

Object

- Definição
- Um objeto no JS é um tipo de dados que possui pares *propriedade : valor*. Podemos ver um objeto como o representante de uma estrutura como um usuário (que tem as propriedades nome, idade, etc…) ou um carro (que tem propriedades marca, modelo, etc…).
- Para declarar um objeto utilizamos as chaves.
    - Exemplo

```
const user = {
  name: 'Juca',
  idade: 27
}
```

- Para acessar suas propriedades temos duas formas: estática e dinâmica.
    - Exemplo:

```
console.log(user)
console.log(user.name)
console.log(user['idade'])
```

- Os nomes de propriedades podem conter espaços e até acentos, mas tem que ser criados como strings.
    - [EXEMPLO]:

```
const user = {
  name: 'Juca',
  idade: 27,
  'data de criação': '15/08/2021',
}
```

- Os nomes das propriedades podem ser criados de forma dinâmica utilizando chaves
    - [EXEMPLO]

```
const newProp = 'data de atualização'
const user = {
  name: 'Juca',
  idade: 27,
  'data de criação': '15/08/2021',
  [newProp]: '15/10/2021',
}
console.log(user[newProp])
```

- Qualquer tipo pode ser o valor de uma propriedade (array e objeto inclusive).
    - [EXEMPLO]

```
const user = {
  name: 'Juca',
  idade: 27,
```

```
    projeto: { name: 'Dev Front-end', prazo: 150 },
    competencias: ['dev python', 'gestão de projetos'],
}
```

- [EXEMPLO] Mostrar acessos no exemplo acima

- Podemos deletar propriedades de um objeto com a instrução delete
  - [EXEMPLO]

```
delete user.competencias
console.log(user)
```

- Assim como variáveis que não atribuímos valores, às propriedades que não declaramos explicitamente no objeto terão valores undefined
  - [EXEMPLO]

```
console.log(user.prop)
```

Referências

- JavaScript Objects (w3schools.com)
- O básico sobre objetos JavaScript - Aprendendo desenvolvimento web | MDN (mozilla.org)

Valores booleanos dos tipos primitivos

- Todos os tipos no JavaScript tem valores booleanos associados.
  - [EXEMPLO]

```
const string = 'Juca'
if (string) {
    console.log('Valor booleano verdadeiro!')
}
```

- Podemos inclusive mudar o tipo de variáveis usando o Cast
  - [EXEMPLO]

```
const string = '12'
console.log(string)
console.log(Number(string))
```

- Para checarmos o tipo de uma variável podemos usar a palavra-chave typeof
  - [EXEMPLO]

```
const string = '12'
console.log(typeof string)
console.log(typeof Number(string))
```

Referências

- Booleano (Boolean) do JavaScript. Nesse tutorial você vai aprender sobre… | by Ricardo Reis | Medium
- Referência prática do operador typeof em JavaScript (devfuria.com.br)

Documentação

- Uma soft skill muito importante para devs é conseguir ler e entender documentações de linguagens, pacotes, frameworks, libs, etc…
- Para a linguagem JS as principais referências que vamos usar são:
    - https://www.w3schools.com
    - https://developer.mozilla.org/pt-BR/
- Vamos usar essas documentações para explorar as classes que o JS tem.

Classe e Instância

- Todos os valores no JS tem um tipo, esse tipo identifica a classe do valor.
- A classe desses valores vai nos permitir realizar operações e funções específicas nesses valores assim como acessar atributos.
    - [EXEMPLO]

```
console.log(10 + 12)
console.log('Hello' + ' World')
```

    - Ambos os exemplos acima utilizam a operação de soma, mas dependendo do tipo de dados o resultado é diferente.
- Com a palavra chave new podemos criar uma nova instância de uma classe, isso é comum quando queremos manipular a classe Date, que representa uma data.

```
const date = new Date()
console.log(date)
```

- Date
    - https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Date
- Number
    - https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Number
- String
    - https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String
- Object
    - https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Object
- Array (básicos sem métodos de repetição)
    - https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array
- É importante sabermos ler a documentação desses métodos, por exemplo, entender seus argumentos e seus retornos. Isso permite que possamos encadear chamadas de métodos ou utilizar os atributos em outras operações.
    - Exemplo:

```
const array = ['banana', 'maçã', 'pêssego']
console.log(array.pop().toUpperCase())
```

Referências

- A diferença entre classes, objetos e instâncias (algerianembassy-kuwait.com)

Template String

- Além das aspas simples e aspas duplas existe uma terceira forma de se criar uma instância de String: com crases.
    - [EXEMPLO]
    - ```
      const string = `Formação Otterwise`
      ```
- Com a template string temos que tomar cuidado com novas linhas e tabulação.
- Mas a principal diferença é que podemos resolver expressões dentro da template string.
    - Exemplo
    - ```
      const num1 = 10
      const num2 = 20
      console.log(`A soma de ${num1} com ${num2} é igual a ${num1 + num2}`)
      ```

Referências

- Como usar template string em JavaScript - Hora de Codar
- Template literals (Template strings) - JavaScript | MDN (mozilla.org)
-

Exercícios

1. Crie um algoritmo que tem como entrada um array de números e imprime no console a soma dos elementos sendo cada um deles multiplicado pelo seu índice.
   Exemplo entrada: [5, 9, 10, 6]
   Exemplo Saída: 47
2. Crie um algoritmo que tem como entrada um objeto e imprime no console os nomes dos projetos ativos do usuário.
   user = {
   name: "Juca",
   projects: [
   { name: "Projeto 1", start: "01/02/2021" , active: true},
   {name: "Projeto 2", start: "03/03/2021", active: false},
   {name: "Projeto 3", start: "10/08/2021", active: true},
   {name: "Projeto 4", start: "20/08/2021", active: false},
   {name: "Projeto 5", start: "18/10/2021", active: true}
   ]
   }
3. Levando em consideração o array [6, 21, 9, 2, 50, 98, 1] crie uma função que mostra o maior numero da lista

4. Crie uma função que conte quantas palavras existem na frase que for passada como parâmetro (dica: utilizem o método split de string)
5. Através do array de usuários abaixo imprima no console todas a skills que cada usuario tem:

```
const users = [
  {
    name: "Joao",
    skills: ["Javascript", "ReactJS", "Redux"]
  },
  {
    name: "Pedro",
    skills: ["VueJS", "Ruby on Rails", "Elixir"]
  }
]
```

6. Crie uma função chamada rockPaperScissorsWinner e recebe dois valores como parâmetro. Esses dois valores podem ser:
    - 0: tesoura
    - 1: papel
    - 2: pedra

    Construa um algoritmo que recebe esses valores randomicamente e printa na tela qual usuário ganhou, se o usuário 1 ou usuário 2