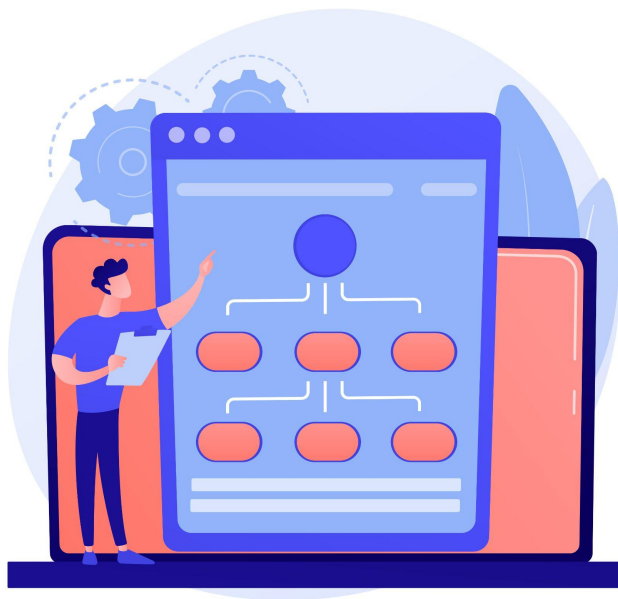


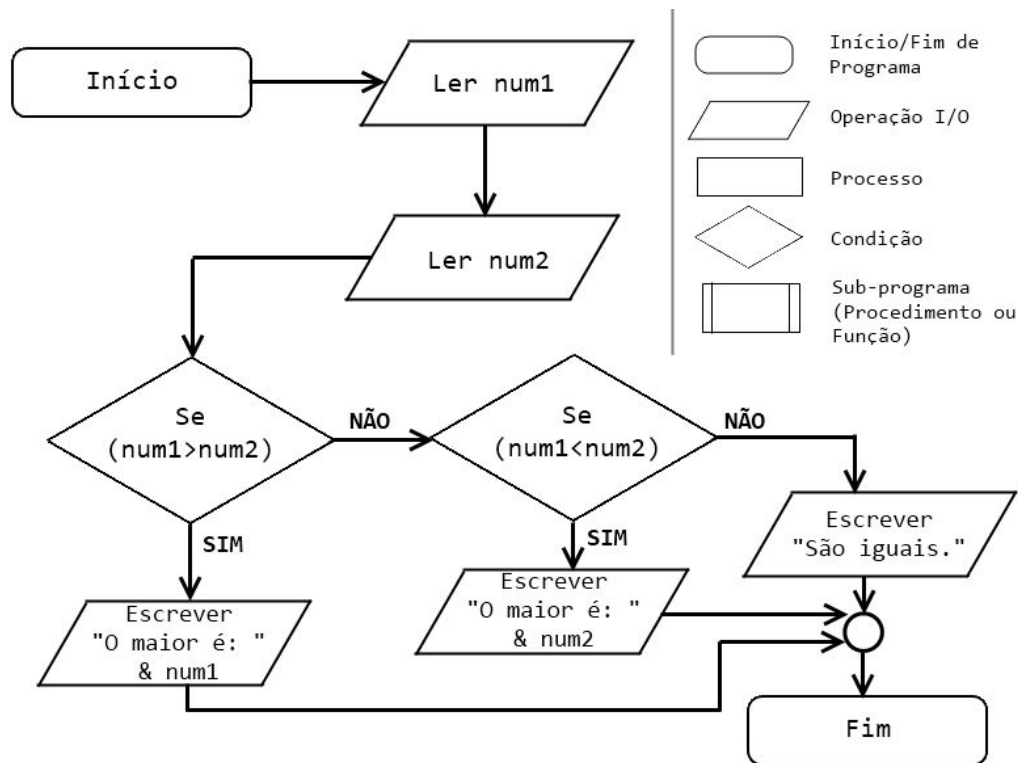
Estruturas de Controle

Introdução



Estruturas de Controle

Introdução



Estruturas de Controle

Introdução

Quais são os principais tipos de estruturas de controle em JavaScript?

Estruturas de Controle

Estruturas condicionais

if/else: Permite executar um bloco de código se uma condição for verdadeira e outro bloco caso seja falsa.

```
//define uma condição simples
if (numero > 0) {
  console.log("O número é positivo.");
}
```

```
// define uma condição de if...else
let idade = 18;
if (idade >= 18) {
  console.log("Você é maior de idade.");
} else {
  console.log("Você é menor de idade.");
}
```

```
//define multiplas condições
const nota = 7;

if (nota >= 9) {
  console.log("Parabéns! Você tirou A.");
} else if (nota >= 7) {
  console.log("Boa! Você tirou B.");
} else if (nota >= 5) {
  console.log("Você tirou C. Estude mais para a próxima!");
} else {
  console.log("Reprovado. É hora de revisar o conteúdo.");
}
```

Estruturas de Controle

Estruturas condicionais

if/else ternário: É uma forma mais curta de escrever uma estrutura condicional if...else.



```

const idade = 18;
const podeDirigir = idade >= 18 ? "Sim" : "Não";
console.log(podeDirigir); // Saída: Sim
    
```

Estruturas de Controle

Operadores lógicos

Quais são os principais operadores lógicos?

Estruturas de Controle

Operadores lógicos

E (&&): Retorna **true** apenas se ambas as expressões forem **true**.



```

if (nomeInformado === nomeUsuario && senhaInformada === senhaCorreta) {
  console.log("Login bem-sucedido!");
} else {
  console.log("Login inválido. Verifique suas credenciais.");
}
  
```

Estruturas de Controle

Operadores lógicos

ou (||): Retorna **true** se pelo menos uma das expressões for **true**.



```

const idade = 16;
const temCarteirinhaEstudante = true;

if (idade >= 18 || temCarteirinhaEstudante) {
  console.log("Você pode retirar livros.");
} else {
  console.log("Desculpe, você não pode retirar livros.");
}
  
```


Estruturas de Controle

Operadores lógicos

NÃO (!): Inverte o valor **booleano** de uma expressão.



```

const idade = 17;

if (!(idade >= 18)) {
  console.log("Você não pode dirigir.");
} else {
  console.log("Você pode dirigir.");
}
  
```

Estruturas de Controle

Estruturas condicionais

switch: Permite escolher entre várias opções com base no valor de uma expressão.



```

let diaSemana = 2;
switch (diaSemana) {
  case 1:
    console.log("Domingo");
    break;
  case 2:
    console.log("Segunda");
    break;
  // ...
}

```

Estruturas de Controle

Estruturas de repetição

O que são estruturas de repetição?

Estruturas de Controle

Estruturas de repetição

Estruturas de repetição: Conhecidas como **loops**, são mecanismos em programação que permitem executar um bloco de código repetidamente enquanto uma determinada condição for verdadeira.

Estruturas de Controle

Estruturas de repetição

Quais são os principais estruturas de repetição em JavaScript?

Estruturas de Controle

Estruturas de repetição

for: Utilizado quando se sabe o número exato de iterações que serão realizadas.



```
let numero = 5;

for (let i = 1; i <= 10; i++) {
  console.log(numero + " x " + i + " = " + (numero * i));
}
```

Estruturas de Controle

Estruturas de repetição

while: Executa um bloco de código enquanto uma condição for verdadeira.



```

let i = 0;
while (i <= 5) {
  console.log(i); // Imprime os números de 0 a 5
  i++;
}
  
```

Estruturas de Controle

Estruturas de repetição

do...while: Similar ao while, mas garante que o bloco de código seja executado pelo menos uma vez.



```

let i = 0;
do {
  console.log(i); // Imprime os números de 0 a 5
  i++;
} while (i <= 5);
  
```


Estruturas de Controle

Estruturas de repetição

Quando usamos estruturas de repetição?



Iteração sobre arrays

Estruturas de Controle

Vetores (arrays)

O que são arrays em JavaScript?

Estruturas de Controle

Vetores (arrays)

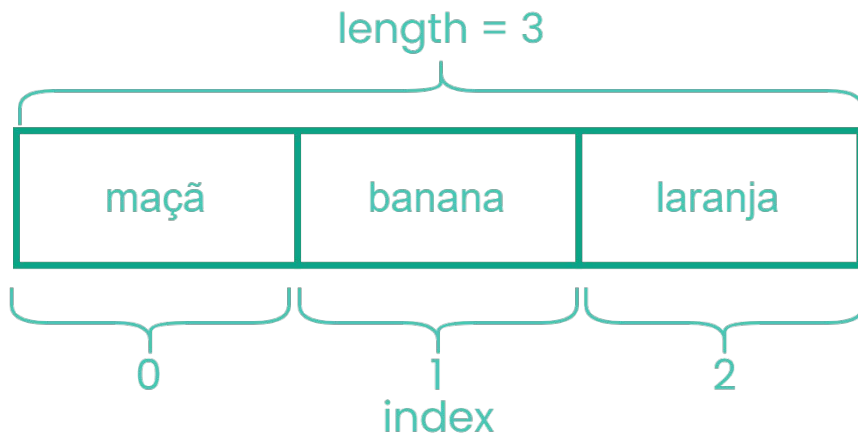
Vetores (arrays): Vetores são estruturas de dados que nos permitem armazenar **múltiplos valores** sob uma **única variável**. Cada elemento pode ser de qualquer tipo de dado (números, strings, objetos, etc.).

```

//declaração
const frutas = ["maçã", "banana", "laranja"];

//acesso:
console.log(frutas[0]); // Imprime: maçã

//modificação
frutas[1] = "pera";
    
```



Estruturas de Controle

Exemplo

Minha Lista


+

Produto 1


Produto 2

Produto 3

```
//declaração
const primeiroProduto = "Produto 1";
const segundoProduto = "Produto 2";
```




```
const produtos = "Produto 1, Produto 2";
const lista = produtos.split(',');
//[ "Produto 1", "Produto 2" ]
```



```
const produtos = [];

produtos.push("Produto 1");
produtos.push("Produto 2");
produtos.push("Produto 2");
```



Estruturas de Controle

Vetores (arrays): Métodos para adicionar e remover elementos

```

const frutas = ['maçã', 'banana'];

//push(): Adiciona um ou mais elementos ao final do array.
frutas.push('laranja');
// frutas = ['maçã', 'banana', 'laranja']

//pop(): Remove e retorna o último elemento do array.
const ultimoElemento = frutas.pop();
// frutas = ['maçã', 'banana']

//shift(): Remove e retorna o primeiro elemento do array.
const primeiroElemento = frutas.shift();
// frutas = ['banana']

//unshift(): Adiciona um ou mais elementos ao início do array.
frutas.unshift('uva');
// frutas = ['uva', 'banana']
  
```

Estruturas de Controle

Vetores (arrays): Métodos para modificar arrays

```

const numeros = [1, 2, 3, 4, 5];

//splice(): Remove ou substitui elementos em um array.
numeros.splice(2, 1); // Remove o elemento no índice 2 (3)

//slice(): Retorna uma nova array com uma parte de um array existente.
const parteDoArray = numeros.slice(1, 3); // [2, 3]

//indexOf(): Retorna o índice do primeiro elemento encontrado.
const indiceNumeroUm = numeros.indexOf(1);

//concat(): Combina dois ou mais arrays em um novo array.
const outros = [6, 7, 8, 9, 10];
numeros.concat(outros);
//[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
  
```

Estruturas de Controle

Estruturas de repetição

```
const frutas = ['maçã', 'banana', 'laranja'];

for (let i = 0; i < frutas.length; i++) {
  console.log(frutas[i]);
}
```

```
const cores = ['vermelho', 'verde', 'azul'];

for (const cor of cores) {
  console.log(cor);
}
```

Estruturas de Controle

Referências

1. IEPSEN, Edécio. Lógica de Programação e Algoritmos com Javascript. 2. ed. São Paulo: Novatec, 2024.
2. SOUZA, Marco. Algoritmos e Lógica de Programação: Um Texto Introdutório para Engenharia. 2. ed. São Paulo: Cengage Learning, 2013.

Palavras-chave

1. if, else, switch, for, while, array