



ESTRUTURAS DE DADOS EM JAVASCRIPT

ESTRUTURA DE DADOS

CST em Desenvolvimento de Software Multiplataforma



Prof. Me. Tiago A. Silva
@prof.tiagotas



INTRODUÇÃO ÀS ESTRUTURAS DE DADOS

- Estruturas de dados são formas organizadas de armazenar e gerenciar informações de maneira eficiente. Elas permitem que os dados sejam manipulados para realizar operações específicas, como inserção, busca, remoção e ordenação. Em linguagens como JavaScript, estruturas de dados desempenham um papel crucial na escrita de código eficiente e na otimização de processos computacionais.
- Estruturas de dados básicas incluem **arrays**, **listas**, **pilhas**, filas, árvores e grafos. Cada uma possui suas características e aplicações particulares.

A IMPORTÂNCIA DAS ESTRUTURAS DE DADOS

- O desempenho de um programa frequentemente depende da escolha correta de uma estrutura de dados. Operações simples, como buscar um item em uma lista ou organizar dados, podem ser muito mais rápidas e eficientes quando a estrutura correta é usada. Além disso, **as estruturas de dados são a base de muitos algoritmos importantes**, como ordenação, busca e processamento de dados.

ARRAYS EM JAVASCRIPT

ARRAYS EM JAVASCRIPT

- Um **array** é uma coleção de itens armazenados em um local de memória contínuo. Em JavaScript, arrays são um tipo especial de objeto usado para armazenar múltiplos valores em uma única variável. Eles permitem acesso rápido a qualquer elemento com base em seu índice, que começa em zero.

```
JS aula4 - exemplos.js X
JS aula4 - exemplos.js > ...
1 let numeros = [1, 2, 3, 4, 5];
2 console.log(numeros[0]); // Saída: 1|
```

COMO DECLARAR E INICIALIZAR ARRAYS

- Declarar arrays em JavaScript é simples e pode ser feito de várias maneiras. O mais comum é utilizar **colchetes []** para criar e inicializar um array:

```
JS aula4 - exemplos.js ×  
JS aula4 - exemplos.js > ...  
1   let frutas = ['maçã', 'banana', 'laranja'];  
2
```

ACESSANDO E MODIFICANDO ELEMENTOS DO ARRAY

- Cada elemento de um **array** é acessado por seu **índice**. Para modificar um elemento, basta referenciar o índice e atribuir um novo valor:

JS aula4 - exemplos.js ×

JS aula4 - exemplos.js > ...

```
1 let cores = ['vermelho', 'verde', 'azul'];
2 cores[1] = 'amarelo'; // Altera o segundo elemento para 'amarelo'
3 console.log(cores); // Saída: ['vermelho', 'amarelo', 'azul']
4
```

MÉTODOS COMUNS DE ARRAYS

- Arrays em JavaScript vêm com diversos métodos embutidos que facilitam a manipulação dos dados armazenados:
- `push()`: Adiciona um ou mais elementos ao final de um array.
- `pop()`: Remove o último elemento de um array.
- `shift()`: Remove o primeiro elemento de um array.
- `unshift()`: Adiciona um ou mais elementos ao início de um array.

MÉTODOS COMUNS DE ARRAYS

JS aula4 - exemplos.js ×

JS aula4 - exemplos.js > ...

```
1 let numeros = [1, 2, 3];
2 numeros.push(4); // [1, 2, 3, 4]
3 numeros.pop(); // [1, 2, 3]
4
```

ITERANDO SOBRE ARRAYS

- JavaScript oferece várias maneiras de percorrer um array, desde laços `for` tradicionais até métodos mais modernos como `forEach` e `map`.

```
JS aula4 - exemplos.js ×
JS aula4 - exemplos.js > ...
1 let numeros = [1, 2, 3, 4];
2 numeros.forEach(num => {
3   console.log(num); // Saída: 1, 2, 3, 4
4 });
```

ARRAYS MULTIDIMENSIONAIS

- Arrays podem conter outros arrays, formando estruturas multidimensionais.

JS aula4 - exemplos.js ×

JS aula4 - exemplos.js > ...

```
1 let matriz = [
2     [1, 2, 3],
3     [4, 5, 6],
4     [7, 8, 9]
5 ];
6 console.log(matriz[0][1]); // Saída: 2
```

PILHAS EM JAVASCRIPT

PILHAS (STACKS) EM JAVASCRIPT

- **Definição de Pilha:**
 - Uma pilha é uma estrutura de dados que segue o princípio **LIFO (Last In, First Out)**, ou seja, o último elemento inserido é o primeiro a ser removido. Isso é similar a uma pilha de pratos, onde o último colocado no topo é o primeiro a ser retirado.
- As pilhas são úteis em diversas situações, como na implementação de algoritmos de "desfazer/refazer" (undo/redo) em editores de texto, na navegação entre páginas da web (histórico de navegação) e na avaliação de expressões aritméticas.
- **Implementando Pilhas com Arrays**
 - Em JavaScript, arrays são usados frequentemente para simular o comportamento de pilhas, uma vez que os métodos `push()` e `pop()` já implementam as operações básicas.

PILHAS (STACKS) EM JAVASCRIPT

JS aula4 - exemplos.js X

JS aula4 - exemplos.js > ...

```
1 let pilha = [];
2 pilha.push(1); // Adiciona 1 à pilha
3 pilha.push(2); // Adiciona 2 à pilha
4 let topo = pilha.pop(); // Remove e retorna o último elemento, 2
```

- Métodos Básicos da Pilha:
 - push(): Adiciona um item no topo da pilha.
 - pop(): Remove o item do topo da pilha.
 - peek(): Retorna o item no topo sem removê-lo.
 - isEmpty(): Verifica se a pilha está vazia.

FILAS EM JAVASCRIPT

FILAS (QUEUES) EM JAVASCRIPT

- **Definição de Fila**
 - Uma fila é uma estrutura de dados que segue o princípio **FIFO (First In, First Out)**, ou seja, o primeiro elemento a ser inserido será o primeiro a ser removido. Um exemplo prático de uma fila seria uma fila de atendimento em um banco.
- **Aplicações Práticas de Filas**
 - Filas são amplamente usadas em sistemas que processam tarefas em ordem sequencial, como impressão de documentos, atendimento de chamadas telefônicas e gerenciamento de processos em sistemas operacionais.

FILAS (QUEUES) EM JAVASCRIPT

- **Implementando Filas com Arrays**
- Em JavaScript, arrays podem ser usados para implementar filas utilizando os métodos `push()` e `shift()`.

JS aula4 - exemplos.js ×

JS aula4 - exemplos.js > ...

```
1 let fila = [];
2 fila.push(1); // Adiciona 1 à fila
3 fila.push(2); // Adiciona 2 à fila
4 let primeiro = fila.shift(); // Remove e retorna o primeiro elemento, 1
```

MÉTODOS BÁSICOS DA FILA

- `enqueue()`: Adiciona um item ao final da fila (usando `push()`).
- `dequeue()`: Remove o primeiro item da fila (usando `shift()`).
- `front()`: Retorna o primeiro item sem removê-lo.
- `isEmpty()`: Verifica se a fila está vazia.

JS aula4 - exemplos.js ×

JS aula4 - exemplos.js > ...

```
1 let fila = [];
2 fila.push('Cliente 1');
3 fila.push('Cliente 2');
4 console.log(fila.front()); // Saída: 'Cliente 1'
```

COMPARAÇÃO ENTRE ARRAY, FILAS E PILHAS

COMPARAÇÃO ENTRE ARRAYS, PILHAS E FILAS

- **Diferenças Fundamentais**

- Arrays: Estruturas de dados lineares que permitem acessar qualquer elemento diretamente através de seu índice.
- Pilhas: Estruturas LIFO, onde o último item adicionado é o primeiro a ser removido.
- Filas: Estruturas FIFO, onde o primeiro item adicionado é o primeiro a ser removido.

COMPARAÇÃO ENTRE ARRAYS, PILHAS E FILAS

- **Quando Usar Cada Estrutura?**
 - Use arrays quando precisar de acesso rápido a qualquer elemento.
 - Use pilhas quando precisar de uma solução para processar dados em ordem reversa (último adicionado, primeiro removido).
 - Use filas quando precisar processar dados em ordem cronológica ou sequencial.

EXERCÍCIOS

EXERCÍCIO 1: ARRAYS

- Escreva uma função em JavaScript que recebe um array de números inteiros e retorna a soma dos números pares do array.

EXERCÍCIO 1: ARRAYS - RESOLUÇÃO

```
JS aula4 - exemplos.js ×
JS aula4 - exemplos.js > ...
1  function somaDosPares(array) {
2      let soma = 0;
3
4      // Percorre o array
5      array.forEach(num => {
6          // Verifica se o número é par
7          if (num % 2 === 0) {
8              soma += num;
9          }
10     });
11
12     return soma;
13 }
14
15 // Exemplo de uso
16 let numeros = [1, 2, 3, 4, 5, 6, 7, 8, 9];
17 console.log(somaDosPares(numeros)); // Saída: 20 (2 + 4 + 6 + 8)
```

EXERCÍCIO 2: PILHAS

- Implemente uma pilha em JavaScript que simule um sistema de desfazer (undo). A pilha deve armazenar ações executadas e permitir desfazer a última ação. O exercício consiste em implementar as funções adicionarAcao para adicionar uma ação à pilha e desfazer para remover e mostrar a última ação.

EXERCÍCIO 2: PILHAS - RESOLUÇÃO

```
JS aula4 - exemplos.js X
JS aula4 - exemplos.js > ...
1  class PilhaDeAcoes {
2      constructor() {
3          this.pilha = [];
4      }
5
6      adicionarAcao(acao) {
7          this.pilha.push(acao);
8          console.log(`Ação adicionada: ${acao}`);
9      }
10
11     desfazer() {
12         if (this.pilha.length === 0) {
13             console.log("Nenhuma ação para desfazer");
14             return;
15         }
16         let acaoDesfeita = this.pilha.pop();
17         console.log(`Ação desfeita: ${acaoDesfeita}`);
18     }
19 }
20
21 // Exemplo de uso
22 let historico = new PilhaDeAcoes();
23 historico.adicionarAcao("Escreveu texto");
24 historico.adicionarAcao("Excluiu palavra");
25 historico.adicionarAcao("Copiou texto");
26
27 historico.desfazer(); // Saída: Ação desfeita: Copiou texto
28 historico.desfazer(); // Saída: Ação desfeita: Excluiu palavra
```

EXERCÍCIO 3: FILAS

- Implemente uma fila de atendimento em JavaScript para um caixa de supermercado. A fila deve permitir adicionar clientes e atender o cliente que está na frente da fila. O exercício consiste em criar as funções `adicionarCliente` para adicionar um cliente à fila e `atenderCliente` para remover o cliente que está sendo atendido.

EXERCÍCIO 3: FILAS - RESOLUÇÃO

```
JS aula4 - exemplos.js X
JS aula4 - exemplos.js > ...
1  class FilaDeAtendimento {
2      constructor() {
3          this.fila = [];
4      }
5
6      adicionarCliente(cliente) {
7          this.fila.push(cliente);
8          console.log(`Cliente adicionado: ${cliente}`);
9      }
10
11     atenderCliente() {
12         if (this.fila.length === 0) {
13             console.log("Nenhum cliente na fila");
14             return;
15         }
16         let clienteAtendido = this.fila.shift();
17         console.log(`Cliente atendido: ${clienteAtendido}`);
18     }
19 }
20
21 // Exemplo de uso
22 let filaSupermercado = new FilaDeAtendimento();
23 filaSupermercado.adicionarCliente("João");
24 filaSupermercado.adicionarCliente("Maria");
25 filaSupermercado.adicionarCliente("Pedro");
26
27 filaSupermercado.atenderCliente(); // Saída: Cliente atendido: João
28 filaSupermercado.atenderCliente(); // Saída: Cliente atendido: Maria
```

EXERCÍCIO COMPLEMENTARES

- **Array:**
 - Escreva uma função que receba um array de palavras e retorne um novo array contendo apenas as palavras que começam com a letra 'A' (maiúscula ou minúscula).
- **Pilha:**
 - Implemente uma pilha que simule a navegação de páginas de um navegador. A pilha deve permitir adicionar uma nova página visitada e voltar para a página anterior (usando voltar). O exercício consiste em implementar as funções navegarPara e voltar.
- **Fila:**
 - Crie uma fila de impressão em JavaScript. A fila deve permitir adicionar documentos para impressão e imprimir o próximo documento na fila. O exercício consiste em implementar as funções adicionarDocumento e imprimirProximo.

OBRIGADO!

- Encontre este **material on-line** em:
 - www.tiago.blog.br
 - Plataforma Teams
- Em caso de **dúvidas**, entre em contato:
 - **Prof. Tiago:** tiago.silva238@fatec.sp.gov.br

