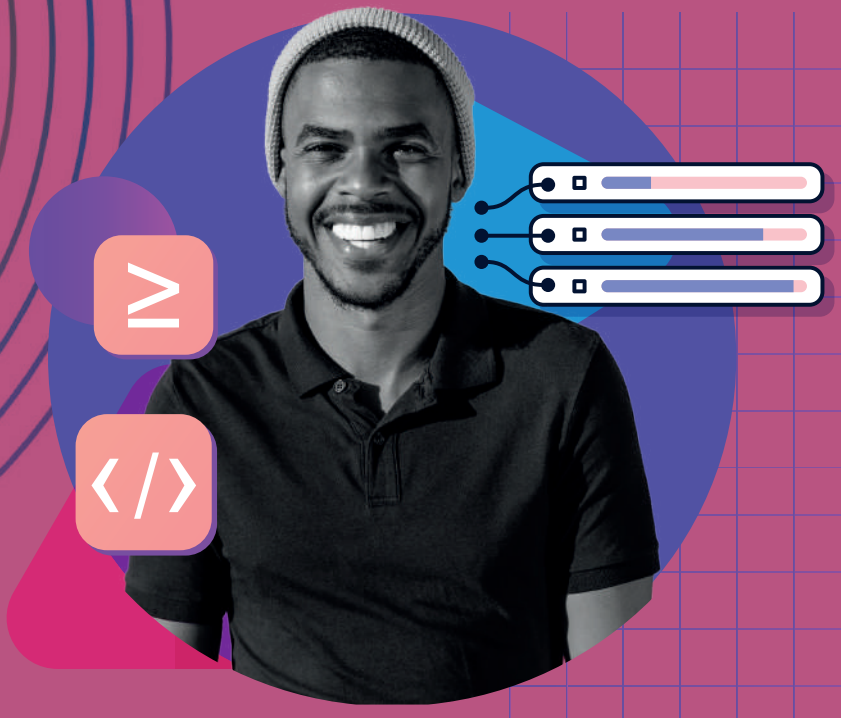
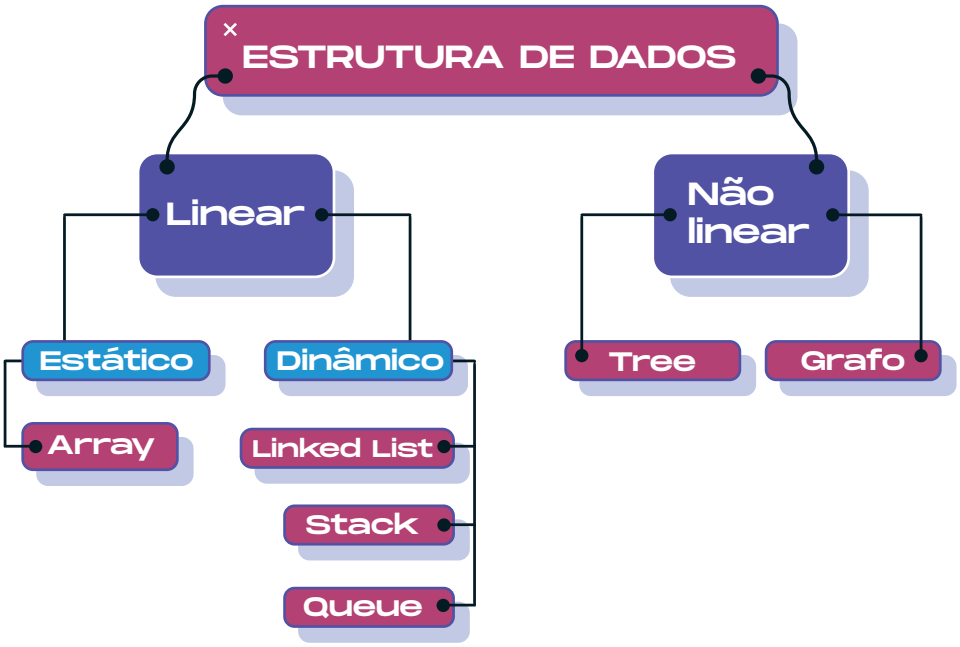
Google olabi

<ESTRUTURADEDADOS NÃOLINEARES>



{ Estrutura de dados não lineares

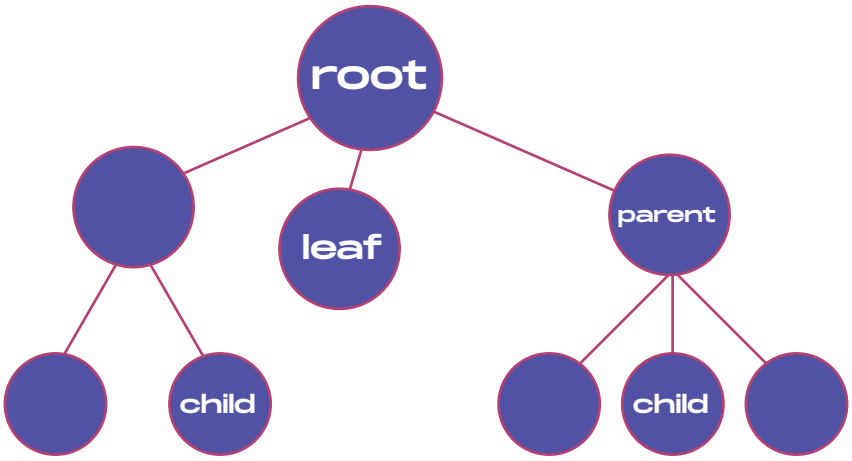
Conforme discutido nas aulas anteriores, estruturas de dados são usadas para armazenar e organizar dados. Também vimos, por meio de algoritmos, como manipular e utilizar essas mesmas estruturas. Diferentes tipos de dados são organizados de forma mais eficiente através do uso de diferentes estruturas. Os tipos abordados até aqui foram: *Arrays*, LinkedLists, *Stack* e *Queues* (estruturas lineares). Nesta aula, iniciaremos o estudo de estruturas de dados não lineares.



{ Árvores

Trees, ou árvores, são estruturas de dados não lineares, frequentemente usadas para representar dados hierárquicos. Esta estrutura apresenta um conjunto de nós ligados por ponteiros, que representam as conexões hierárquicas entre nós. Um nó pode conter dados de qualquer tipo, mas todos os nós precisam fazer parte do mesmo tipo.

O nó mais alto de uma árvore é chamado de “raiz” (*root*, em inglês). Cada elemento diretamente abaixo de um elemento é chamado de “filho” (*child*, em inglês). O elemento intermediário é chamado de “pai” (*parent*, em inglês). O elemento sem filho é chamado de “folha” (*leaf*, em inglês).



Outros componentes desta estrutura de dados:

• ***Sub Árvore:*** árvore menor mantida dentro de uma maior. A raiz dessa árvore pode vir de qualquer nó da árvore maior;
• ***Profundidade:*** número de ponteiros entre o nó e a raiz;
• ***Altura:*** número de ponteiros do caminho mais longo entre um nó e um nó de folha;
• ***Grau:*** número de sub árvores.

<Por que usar árvores?>

As árvores formam uma das organizações mais básicas dos computadores. Sua estrutura hierárquica lhe confere propriedades únicas de armazenamento, manipulação e acesso a dados.

Possíveis usos de uma árvore incluem:

• Armazenamento como hierarquia (por exemplo, sistemas de arquivos de um computador);
• Armazenamento de informações que desejamos pesquisar rapidamente (árvores AVL e Red-Black, por exemplo, foram projetadas para uma busca rápida);
• Herança, analisador XML, aprendizado de máquinas e DNS.

Tipos avançados de árvores como B- Trees e B+ Trees podem ser usados para indexação de um banco de dados. Árvores também são ideais para redes sociais e jogos de xadrez por computador. Uma *Spanning Tree*, por exemplo, pode ser usada para encontrar caminhos mais curtos em roteadores para *networking*.

{ Tipos de árvores

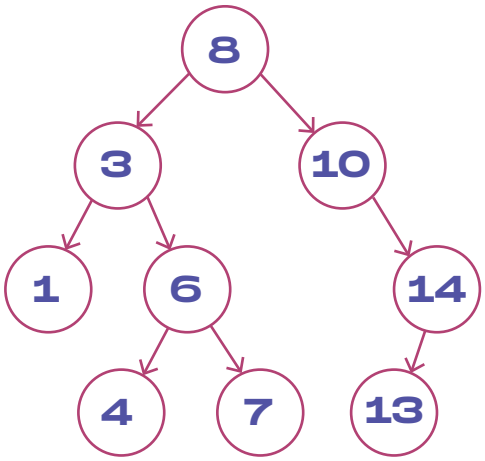
Dentre os diferentes tipos de árvores, as binárias são especiais por serem muito utilizadas em diversas aplicações e porque, quando ordenadas, permitem que pesquisas, inclusões e exclusões de dados em sua estrutura sejam feitas de forma extremamente rápida.

As propriedades de uma árvore de busca binária incluem:

• todos os elementos na sub árvore esquerda de um determinado nó n menores que n;

• todos os elementos na sub árvore direita de um determinado nó n maiores ou iguais a n.

<Exemplo de árvore binária>



<Exemplos de métodos utilizados em árvores binárias>

insert()	Acrescenta um novo nó à árvore
add()	Adiciona o elemento especificado a este conjunto caso ele ainda não esteja presente
isEmpty()	Retorna verdadeiro se o conjunto não contiver elementos
remove()	Remove o elemento especificado do conjunto se ele estiver presente

<Percursos em árvores>

Ordem de percurso	Primeira visita	Segunda visita	Terceira visita
Em ordem	Visitar sub árvore esquerda em ordem	Visitar nó raiz	Visitar sub árvore direita em ordem
Pré ordem	Visitar nó raiz	Visitar sub árvore esquerda em pré-ordem	Visitar sub árvore direita em pré-ordem
Pós ordem	Visitar sub árvore esquerda em pós-ordem	Visitar sub árvore direita em pós-ordem	Visitar nó raiz

Referências

https://www.educative.io/blog/data-structures-trees-java

https://www.devmedia.com.br/trabalhando-com-arvores-binarias-em-java/25749

https://www.javatpoint.com/binary-tree-java

Google olabi

<ESTRUTURADEDADOS>

/exercícios/

{ Exercício coletivo I

Considere uma árvore binária e determine se ela está equilibrada em termos de altura. Neste problema, uma árvore binária com equilíbrio de altura é definida como uma árvore na qual as alturas das sub árvores da esquerda e direita de cada nó diferem até 1, no máximo.

{ Exercício coletivo II:

Considere a raiz de uma árvore binária e retorne a passagem de nível dos valores de seus nós (da esquerda para a direita, nível a nível).

<Extras>

I - Considere uma matriz ordenada de forma crescente e escreva um algoritmo para criar uma árvore binária de altura mínima;

II - Projete um algoritmo para encontrar o primeiro ancestral comum entre dois nós de uma árvore binária evitando o armazenamento de nós adicionais em uma mesma estrutura de dados.

<Leitura recomendada>

MLA. McDowell, Gayle Laakmann, 1982-. Cracking the Coding Interview : 150 Programming Questions and Solutions. Palo Alto, CA :CareerCup, LLC, 2011.

<Referência para os exercícios>

https://leetcode.com/problems/

Professora: Renata Pereira | Monitora: Jessica Osko