Professor
Edson Pedro FERLIN

Estruturas de Dados

Árvores

Prof. Edson Pedro Ferlin

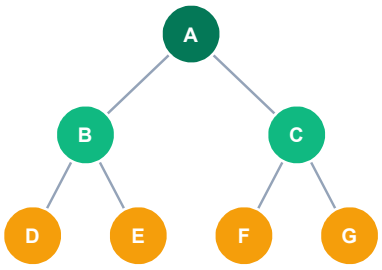
Professor
Edson Pedro FERLIN

Estruturas de Dados

AULA 3 – INTRODUÇÃO ÀS ARVORES

Introdução às Árvores

Definições • Representações • Árvores Binárias • Percursos



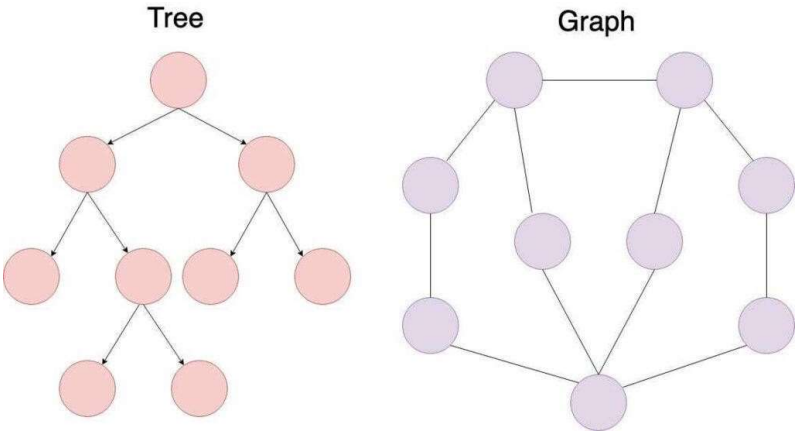
As estruturas sequenciais são suficientes para modelar qualquer tipo de problema?

A resposta é NÃO

- Como você organizaria em memória estruturas como:
  - Um sistema de arquivos (pastas)?
  - Uma hierarquia de empresa?
  - Um ranking de jogos?
- Estruturas lineares não são adequadas para representar relações entre seus elementos, como por exemplo a hierarquia
- **Para modelar de maneira mais adequada esses problemas complexos precisamos de uma estrutura não linear**

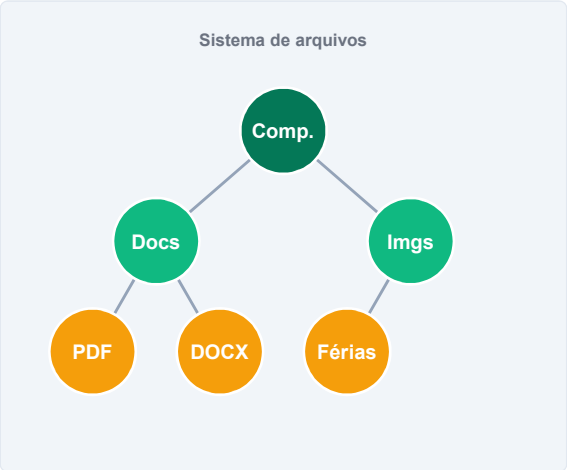
Exemplos de estruturas de dados não lineares

- Árvores
- Grafos
- Dicionários
- Heap
- etc



Onde encontramos árvores?

- **Sistema de arquivos**
- **Organograma empresarial**
- **Árvore genealógica**
- **DOM de páginas HTML**
- **Inteligência Artificial**
- **Compiladores**
- **Bancos de dados**

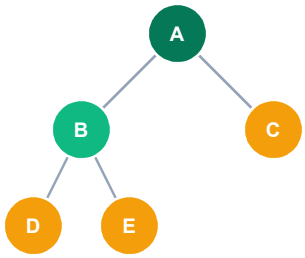


O que é uma Árvore?

Uma árvore é uma estrutura de dados **não linear**, composta por um conjunto de nós conectados por arestas.

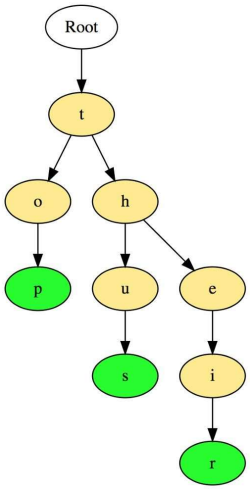
Características

- **Existe apenas um nó raiz.**
- **Todo nó possui zero ou mais filhos.**
- **Não existem ciclos.**
- **Existe um único caminho entre dois nós.**



Árvore

- É a estrutura não linear mais simples
- Permite tratamento computacional para modelar diversos problemas práticos de forma eficiente e simples
- **Definição:** É uma estrutura de dados que armazena as informações de maneira hierárquica, tal que:
  - Com exceção de sua raiz (elemento do topo), todos os demais elementos possuem um elemento pai e 0 ou mais elementos filho.



Tipos Árvores

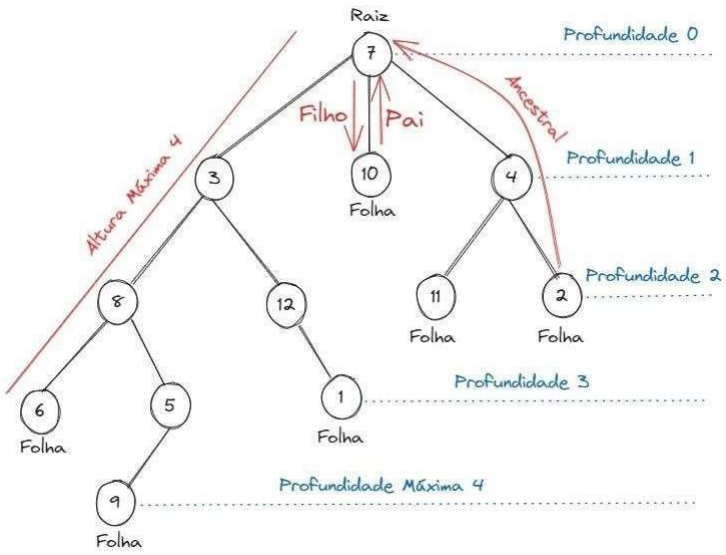
- Existem diferentes tipos de árvores que são especializadas para otimizar um determinado tipo de operação do mundo real
- Exemplos:
  - Árvores Binárias
  - Árvores AVL
  - Árvores Binárias de Busca
  - Árvores Rubro-Negras
  - Árvores B

Definição Matemática

- Uma árvore enraizada T, ou simplesmente árvore, é um conjunto finito de elementos denominados nós ou vértices tais que
  - T = Ø, e a árvore é dita vazia, ou
  - existe um nó especial chamado raiz de T(r(T)); os restantes constituem um único conjunto vazio ou são divididos em m ≥ 1 conjuntos disjuntos não vazios, as subárvores de r(T), ou simplesmente subárvores, cada qual, por sua vez, uma árvore.

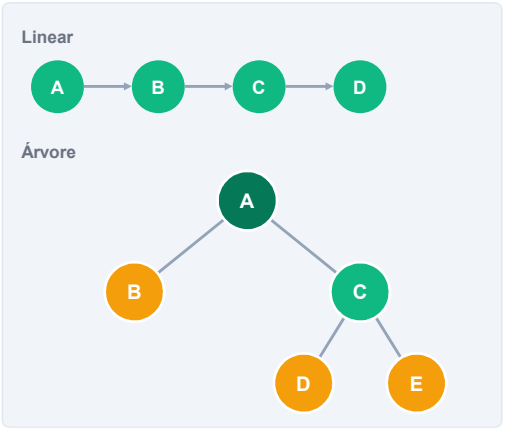
Conceitos

- Floresta: Conjunto de árvores
- Nó: Elemento da árvore
- Raiz: Único nó sem pai
- Folha: Nó sem filhos
- Aresta: Relacionamento/Ligação entre dois nós
- Nó pai/filho do nó: Nó mais próximo acima/abaixo do nó
- Subárvore: árvore cuja raiz é um nó qualquer da árvore principal
- Altura: Distância máxima (número de arestas) entre o nó raiz e o nó folha mais distante



Estruturas Lineares × Não Lineares

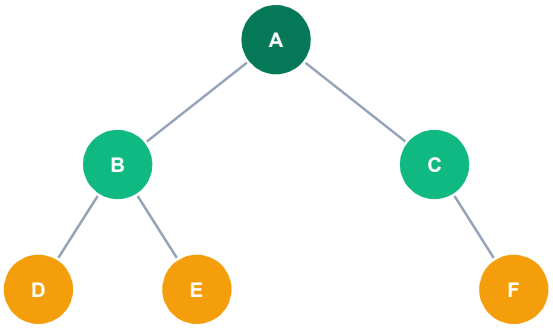
Estruturas Lineares	Árvores
Vetores	Hierarquia
Listas	Ramificações
Pilhas	Diversos níveis
Filas	Organização em níveis



Definições Fundamentais

1. **Grau** — Número de filhos de um nó
2. **Altura** — Maior distância da raiz até uma folha
3. **Profundidade** — Distância de um nó até a raiz
4. **Nível** — Conjunto de nós à mesma profundidade

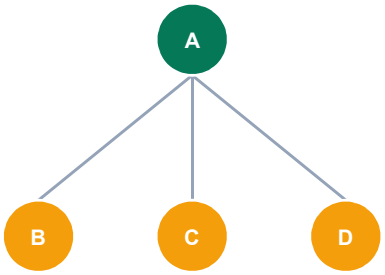
Exemplo



Responda

- **Quem é a raiz?**
- **Quantas folhas existem?**
- **Quem é o pai de E?**

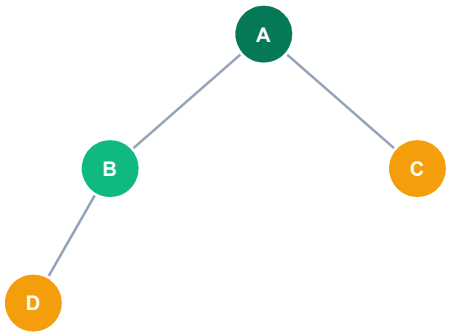
Grau de um Nó



Grau(A) = 3

A raiz possui três filhos: B, C e D.

Altura da Árvore



Altura = 3

Maior número de arestas da raiz A até a folha mais profunda (D).

Representações de Árvores

Ligada

(ponteiros)

- Cada nó aponta para seus filhos
- Cresce dinamicamente
- Uso flexível de memória

Vetorial

(arranjo)

- Nós armazenados por índice
- Filhos calculados por fórmula
- Ideal para árvores completas

Hierárquica

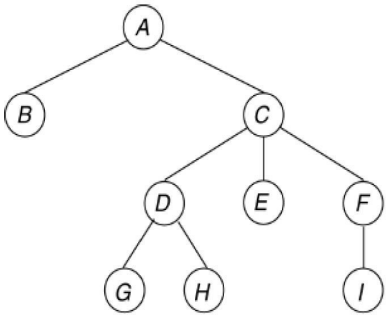
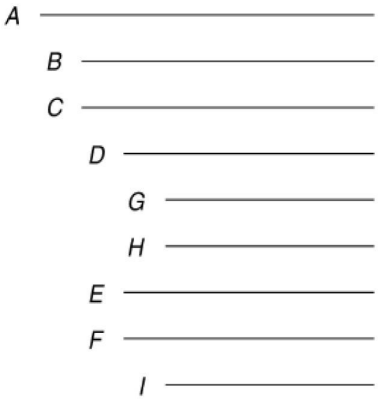


Diagrama de Barras



Parênteses aninhados

(A (B) (C (D (G) (H)) (E) (F (I))))

Representação Ligada

Cada nó guarda o dado e dois ponteiros que referenciam as subárvores esquerda e direita. Quando não há filho, o ponteiro é nulo (None).

Representação em Vetor

valor	A	B	C	D	E	F	G
índice	0	1	2	3	4	5	6

Filho esquerdo

2·i + 1

Filho direito

2·i + 2

Atividade

Para a árvore apresentada em aula, calcule:

1

Altura

da árvore

2

Folhas

quantidade

3

Grau

de cada nó

Árvores Binárias

Cada nó com no máximo dois filhos

Árvore Binária — Definição

Cada nó possui **no máximo dois filhos**.

E

Filho esquerdo

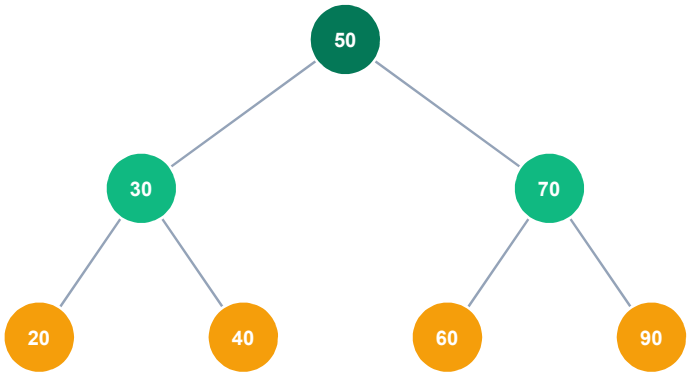
left child

D

Filho direito

right child

Exemplo de Árvore Binária



Árvore binária de busca: valores menores à esquerda, maiores à direita.

Implementação Básica

- Cada nó deve conter seu valor e dois nós filhos (esquerdo e direito) que podem estar vazios (None);
- A classe árvore deve ser instanciada através do nó raiz e deve possuir seus métodos, como percurso em pré-ordem, pós-ordem, em ordem, inserção, remoção, etc.

```
class Node:
  def __init__(self, value):
    self.value = value
    self.left = None
    self.right = None
```

Implementação em Python

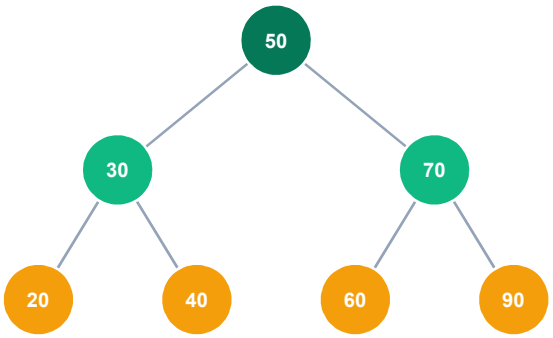
```python
class No:
    def __init__(self, valor):
        self.valor = valor
        self.esquerda = None
        self.direita = None
```

A classe No guarda o valor e as referências para as subárvores esquerda e direita.

Criando a Árvore

```python
raiz = No(50)
raiz.esquerda = No(30)
raiz.direita  = No(70)
raiz.esquerda.esquerda = No(20)
raiz.esquerda.direita  = No(40)
raiz.direita.esquerda  = No(60)
raiz.direita.direita   = No(90)
```

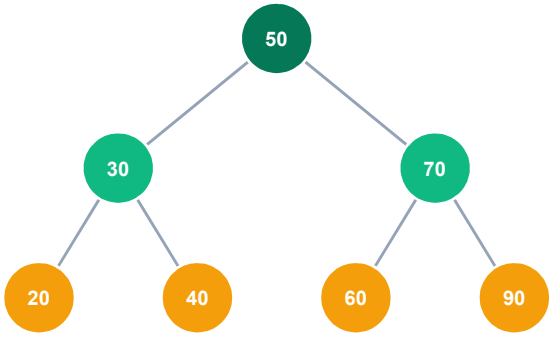
Visualização




```
raiz = No(50)
raiz.esquerda = No(30)
raiz.direita = No(70)
...
```

Cada chamada No(...) cria um nó; as atribuições ligam pais e filhos, formando a árvore ao lado.

Inserção



Novo nó

- **Compara o novo valor com a raiz.**
- **Vai à esquerda se menor, à direita se maior.**
- **Repete até encontrar a posição vazia correta.**

Busca

Buscar o valor 40

- 40 < 50 → vai para a esquerda.
- 40 > 30 → vai para a direita.
- 40 = 40 → valor encontrado!

Aplicações das Árvores Binárias

1. Sistemas Operacionais
2. Compiladores
3. Bancos de Dados
4. Inteligência Artificial
5. Compressão Huffman

Percursos

Visitar todos os nós de forma ordenada

O que é Percurso?

- É o caminho para percorrer todos os nós de uma árvore
- Diferente das estruturas lineares, em árvores temos múltiplos caminhos
- Percursos principais
  - Pré-ordem
  - Em ordem
  - Pós-ordem

Implementação de Percurso

- Os percursos se tratam de definições recursivas entre as subárvores da árvore, dessa forma a maneira mais simples de implementa-los é com uma função recursiva

```
def inorder(node):
  if node:
    inorder(node.left)
    print(node.value)
    inorder(node.right)
```

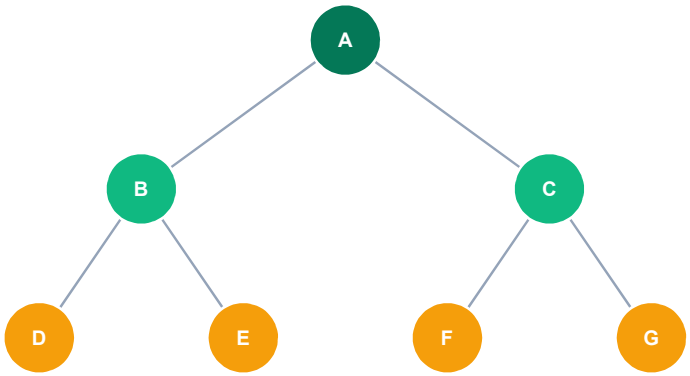

Percursos

Percorrer significa visitar todos os nós da árvore.

Principais métodos

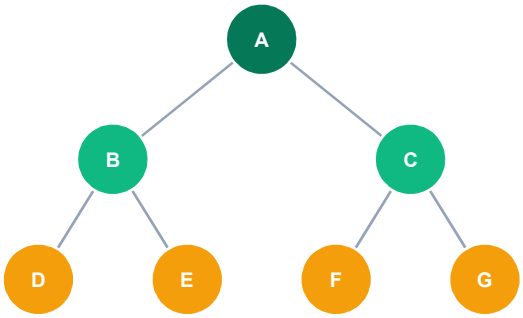
Pré-Ordem	Em Ordem	Pós-Ordem
Raiz → Esq → Dir	Esq → Raiz → Dir	Esq → Dir → Raiz

Árvore de Referência



Usaremos esta árvore em todos os percursos a seguir.

Percurso Pré-Ordem



Ordem

1 · Raiz
2 · Esquerda
3 · Direita

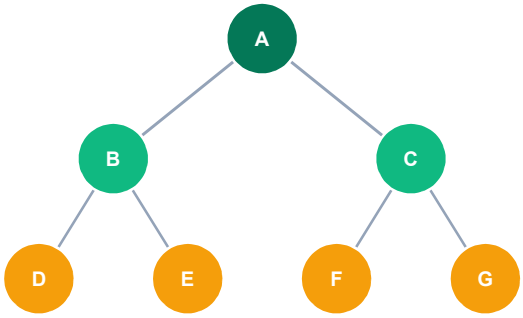
Resultado

A B D E C F G

Pré-Ordem — Python

```python
def pre_ordem(no):
    if no:
        print(no.valor)
        pre_ordem(no.esquerda)
        pre_ordem(no.direita)
```


Percurso em Ordem



Ordem

1 · Esquerda
2 · Raiz
3 · Direita

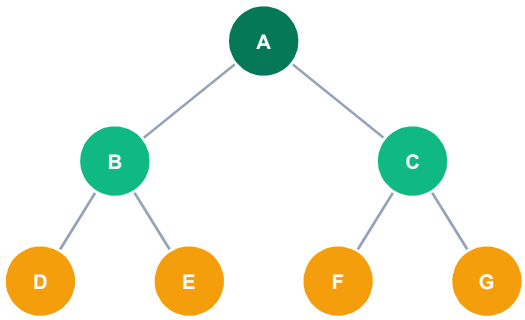
Resultado

D B E A F C G

Percurso em Ordem — Python

```python
def em_ordem(no):
    if no:
        em_ordem(no.esquerda)
        print(no.valor)
        em_ordem(no.direita)
```


Percurso Pós-Ordem



Ordem

1 · Esquerda
2 · Direita
3 · Raiz

Resultado

D E B F G C A

Percurso Pós-Ordem — Python

```
def pos_ordem(no):
    if no:
        pos_ordem(no.esquerda)
        pos_ordem(no.direita)
        print(no.valor)
```

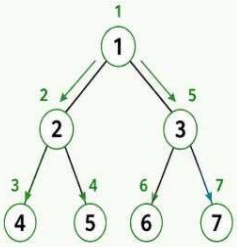

Comparação dos Percursos

Percurso	Ordem de visita
Pré-Ordem	Raiz → Esq → Dir
Em Ordem	Esq → Raiz → Dir
Pós-Ordem	Esq → Dir → Raiz

PRÉ-ORDEM

Raiz → Esquerda → Direita

Visita o nó, depois a subárvore da esquerda e por último a subárvore da direita.

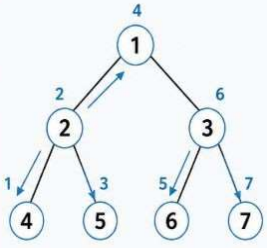


Ordem de visita:
1 → 2 → 4 → 5 → 3 → 6 → 7

EM ORDEM

Esquerda → Raiz → Direita

Visita a subárvore da esquerda, depois o nó e por último a subárvore da direita.

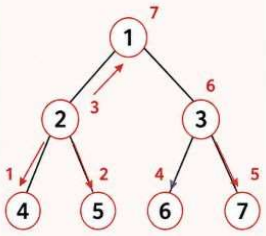


Ordem de visita:
4 → 2 → 5 → 1 → 6 → 3 → 7

PÓS-ORDEM

Esquerda → Direita → Raiz

Visita a subárvore da esquerda, depois a subárvore da direita e por último o nó.



Ordem de visita:
4 → 5 → 2 → 6 → 7 → 3 → 1

Complexidade dos Percursos

Todos os percursos executam em

O(n)

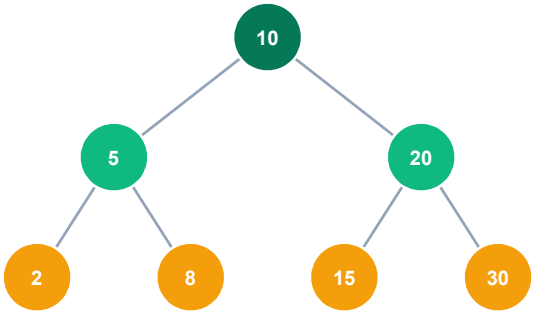
Cada nó é visitado exatamente uma vez.

Atividade - Árvores

Para a árvore apresentada, determine os três percursos:

Pré-Ordem	Em Ordem	Pós-Ordem
_ _ _ _ _ _ _	_ _ _ _ _ _ _	_ _ _ _ _ _ _

Atividade - Árvores



Tarefa

- Construa a árvore acima em Python.
- Execute o percurso Pré-Ordem.
- Execute o percurso Em Ordem.
- Execute o percurso Pós-Ordem.

Atividade - Árvores

Modifique o programa anterior para calcular:

1. **Nós**
   quantidade total
2. **Altura**
   da árvore
3. **Folhas**
   quantidade

Comparação Geral das Estruturas

Estrutura	Organização	Acesso
Vetor	Linear	Índice
Lista	Linear	Sequencial
Pilha	Linear	Topo
Fila	Linear	Extremidades
Árvore	Hierárquica	Percursos

Aplicações

- Navegadores (DOM)
- Árvores Genealógicas
- Sistemas de Arquivos
- Compiladores
- Jogos
- Inteligência Artificial
- Bancos de Dados (B-trees)
- Índices de Busca

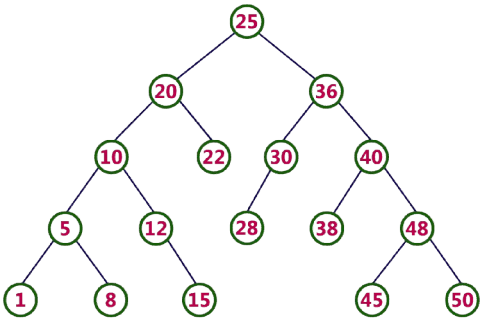
Exercícios

1) Implemente os métodos de busca em pré-ordem e pós-ordem na classe de árvore binária.

2) Implemente um método que conta a quantidade de nós em uma árvore binária.

3) Considere a seguinte árvore binária ao lado, qual seria o resultado das seguintes operações:

a) inserir 42
b) remover 15
c) remover 20
d) inserir 7
e) remover 40



4) Com relação a árvore do exercício 3, apresente o seu percurso em pré-ordem, pós-ordem e em ordem.

Problema

Implemente a classe ArvoreBinaria em Python

- inserir(valor)
- buscar(valor)
- pre_ordem()
- em_ordem()
- pos_ordem()
- contar_nos()
- contar_folhas()
- altura()

Desafio Extra

Implemente uma **Árvore Binária de Busca (BST)**, com inserções automáticas conforme a ordem dos valores. Compare o desempenho de busca com o de uma lista linear.

Contato

eferlin@live.com

(BLOG) professorferlin.blogspot.com

(SITE) professorferlin.com.br

(YOUTUBE) ProfEdsonPedroFerlin