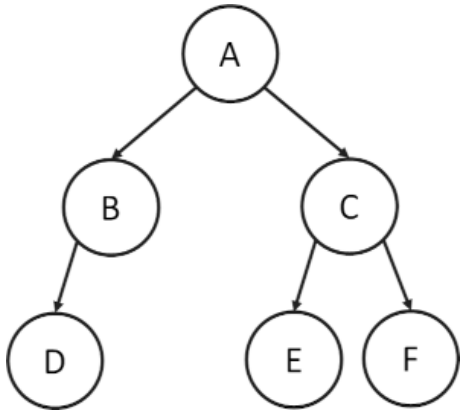
Estrutura de Dados II
Ciência da Computação

Prof. André Kishimoto
2023

Árvore Binária

- *Binary tree* é uma especialização de árvore.
- Uma regra principal:
  - **Cada nó da árvore binária tem zero, um ou dois filhos (no máximo).**
  - Filho esquerdo (*left child*) e filho direito (*right child*).

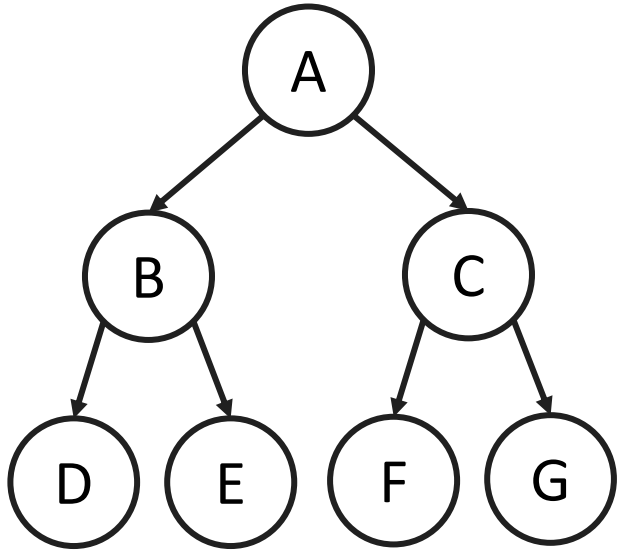


Árvore Binária

Propriedades:

- A **altura** de uma árvore com ***n*** elementos (n > 0) é no **mínimo** $\lfloor \log_2 n \rfloor$ e no **máximo *n-1***.
  - $\log_2 n = x$ (inverso é $2^x = n$)

- Uma árvore binária **não vazia** com altura ***h*** tem no **mínimo** ***h+1*** nós e no **máximo** $2^{h+1} - 1$ nós.



Árvore binária cheia

- Altura mínima e máxima?
- Quantidade mínima e máxima de nós?
- Exemplo de altura máxima?

Árvore Binária

- Podemos definir uma árvore binária de maneira recursiva.
- Uma árvore binária ou é vazia ou consiste em:
  - Um nó raiz da árvore T que armazena um elemento;
  - Uma árvore binária, chamada de subárvore da esquerda de T;
  - Uma árvore binária, chamada de subárvore da direita de T.

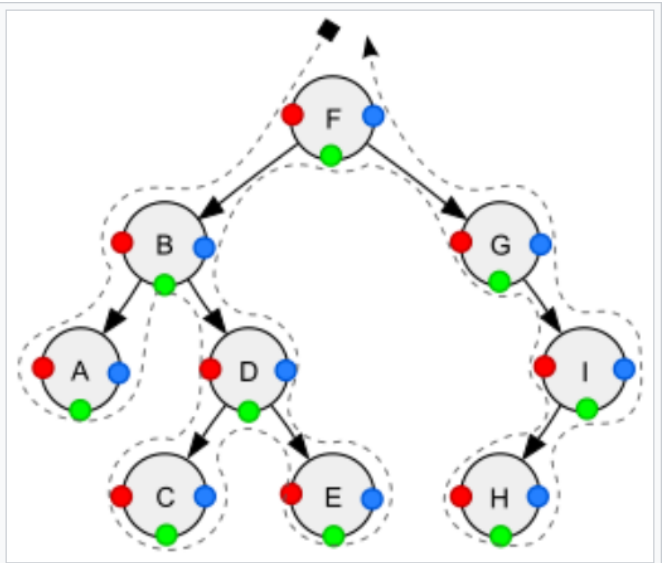
Percurso/travessia em árvore

- Árvores binárias são usadas principalmente para busca.
- É possível percorrer uma árvore:
  - **Pré-ordem:** Visitar a raiz, depois a subárvore esquerda e, por último, a subárvore direita.
  - **Em ordem:** Visitar a subárvore esquerda, depois a raiz e, por último, a subárvore direita.
  - **Pós-ordem:** Visitar a subárvore esquerda, depois a subárvore direita e, por último, a raiz.
  - **Por nível:** Visitar a raiz, depois os nós do próximo nível, da esquerda para a direita, depois os nós do próximo nível, da esquerda para a direita, até percorrer o último nível.

Percurso/travessia em árvore

- Árvores binárias são usadas principalmente para busca.
- É possível percorrer uma árvore:
  - **Pré-ordem:** pré-fixado, pre-order, **NLR** traversal.
  - **Em ordem:** infixado, in-order, **LNR** traversal.
  - **Pós-ordem:** pós-fixado, post-order, **LRN** traversal.
  - **Por nível:** level-order traversal.

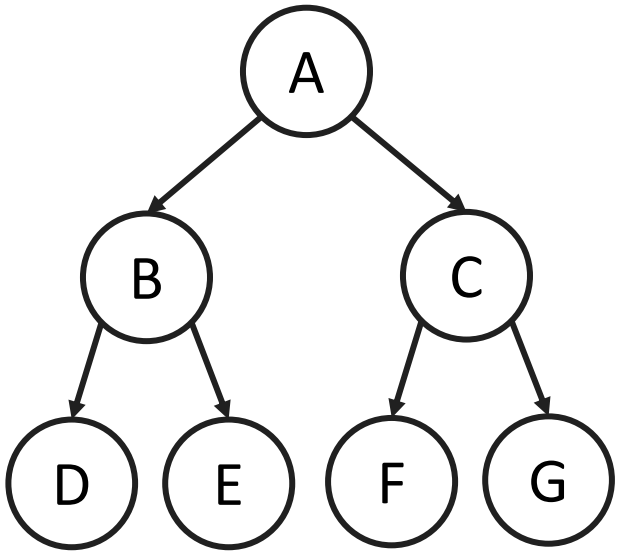
  - N: Node
  - L: Left
  - R: Right



Depth-first traversal (dotted path) of a binary tree:
Pre-order (node visited at position red ●):
F, B, A, D, C, E, G, I, H;
In-order (node visited at position green ●):
A, B, C, D, E, F, G, H, I;
Post-order (node visited at position blue ●):
A, C, E, D, B, H, I, G, F.

https://en.wikipedia.org/wiki/Tree_traversal

Percurso/travessia em árvore



- Percurso em pré-ordem?
- Percurso em ordem?
- Percurso em pós-ordem?
- Percurso por nível?

Percurso/travessia em árvore

- Percurso em pré-ordem?
- Percurso em ordem?
- Percurso em pós-ordem?
- Percurso por nível?

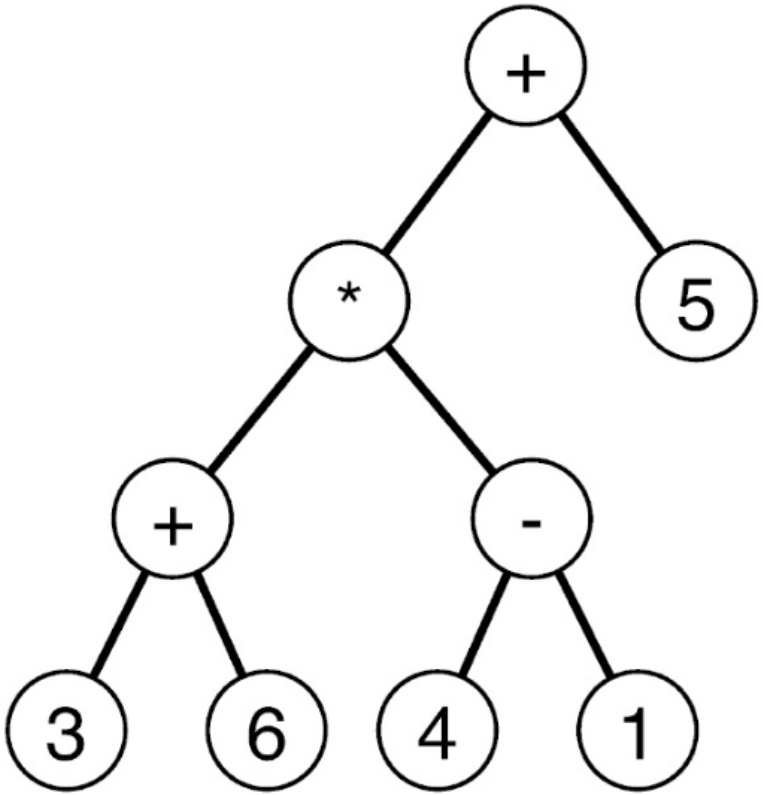


Figura 16.2: Árvore da expressão: $(3 + 6) * (4 - 1) + 5$.

CELES, W.; CERQUEIRA, R.; RANGEL, J. L. Introdução a estrutura de dados: com técnicas de programação em C, 2ª ed. Rio de Janeiro: Elsevier, 2016.

Percurso (...)

Avaliação de uma árvore de expressão

O caminhamento pós-fixado de uma árvore binária pode ser usado para resolver o problema de avaliação de expressões. Nesse problema, dada a árvore de uma expressão aritmética, ou seja, uma árvore binária na qual para cada nodo externo existe um valor associado e para cada nodo interno se associa um operador aritmético (ver Exemplo 7.9), deseja-se calcular o valor da expressão aritmética representada pela árvore.

O algoritmo evaluateExpression, indicado no Trecho de Código 7.24, avalia a expressão associada com a subárvore com raiz no nodo v de uma árvore T, que representa uma expressão aritmética, executando um caminhamento pós-fixado T que se inicia em v. Nesse caso, a ação "de visita" sobre cada nodo consiste na execução de uma operação aritmética simples. Observa-se que se explora o fato de que uma árvore de expressão aritmética é uma árvore binária própria.

```
Algoritmo evaluateExpression(T,v):
  se v é um nodo interno de T então
    seja ○ o operador armazenado em v
    x ← evaluateExpression(T,T.left(v))
    y ← evaluateExpression(T,T.right(v))
    retorna x ○ y
  senão
    retorna o valor armazenado em v
```

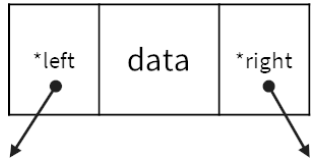
Trecho de Código 7.24 Algoritmo evaluateExpression para calcular a expressão representada pela subárvore de uma árvore T que representa uma expressão aritmética, enraizada no nodo v.

A aplicação de caminhamento pós-fixado na avaliação de expressões aritméticas resulta em um algoritmo que executa em tempo $O(n)$ para avaliar uma expressão aritmética representada por uma árvore binária de n nodos. Na verdade, da mesma forma que o caminhamento pós-fixado genérico, o caminhamento pós-fixado para árvores binárias pode ser aplicado para outros problemas "bottom-up" (como, por exemplo, o problema de cálculo do tamanho apresentado no Exemplo 7.7).

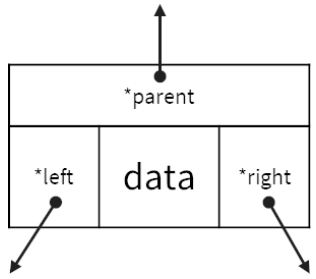
GOODRICH, M. T.; TAMASSIA, R. Estrutura de Dados e Algoritmos em Java, 5ª ed. Porto Alegre: Bookman, 2013.

Implementação

- Uma árvore é composta por um conjunto de nós.
- No caso de uma árvore binária, o mínimo que o nó precisa ter é:
  - A parte dos **dados**, o conteúdo que é armazenado por cada nó (*data*);
  - Um ponteiro para o **nó filho esquerdo** (**left*);
  - Um ponteiro para o **nó filho direito** (**right*).



- Podemos incluir um ponteiro que aponta para o **nó pai** (***parent*).
  - Não é obrigatório, mas ter acesso ao nó pai pode facilitar algumas operações.



Implementação