SCC-223 Estruturas de Dados I

Árvores Binárias

Profa. Elaine Parros Machado de Sousa

ICMC USP SÃO CARLOS

Aula baseada em material do professor Rudinei Goularte.

Grupo de Bases de Dados e Imagens
USP - São Carlos

Árvore Binária- Definição

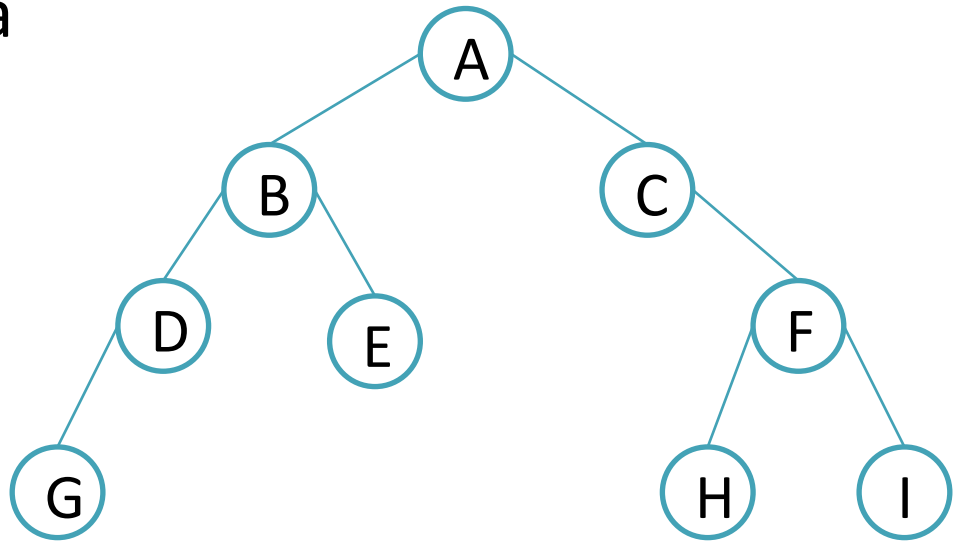
- Uma **Árvore Binária** (AB) **T** é um conjunto finito de elementos, denominados **nós** ou vértices, tal que

  1. Se **T = Ø**, a árvore é dita vazia, ou
  2. **T** contém um nó especial **r**, chamado **raiz** de **T**, e os demais nós podem ser subdivididos em dois sub-conjuntos distintos T_E e T_D, os quais também são árvores binárias (possivelmente vazias)

     - T_E e T_D são denominadas **subárvore esquerda** e **subárvore** direita de **T**, respectivamente

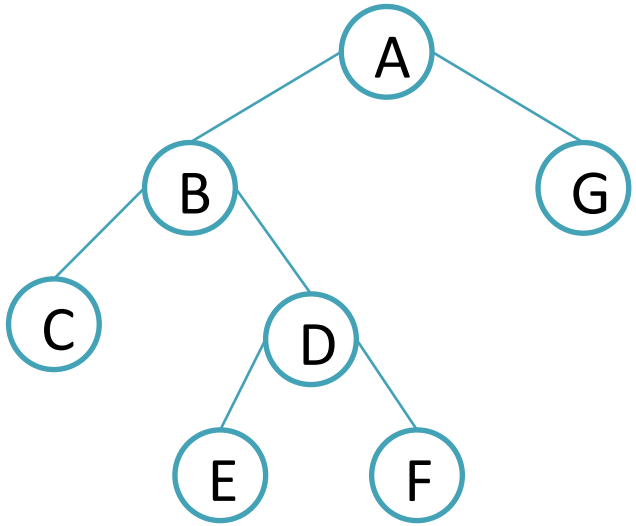
Árvore Binária

- A raiz da subárvore esquerda de um nó **v**, se existir, é denominada **filho esquerdo** de **v**
- Definição análoga para **filho direito** de **v**
- Pela definição da árvore binária, o filho esquerdo pode existir sem o direito, e vice-versa



Árvore Estritamente Binária

- Uma **Árvore Estritamente Binária** (ou Árvore Própria) tem nós com 0 (nenhum) ou 2 (dois) filhos
  - nós interiores (não folhas) sempre têm 2 filhos



Árvore Binária Completa

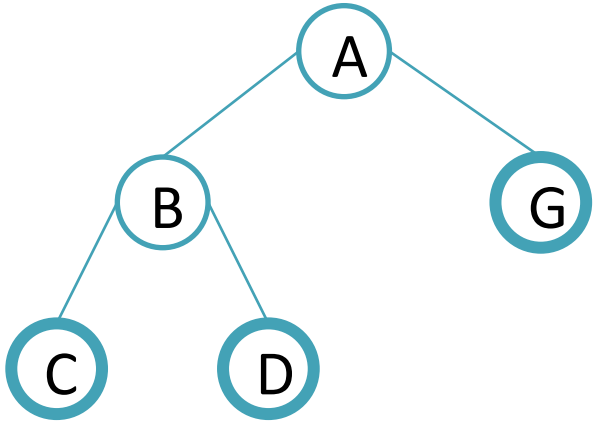
- **Árvore Binária Completa** (ABC)

  Se a **profundidade** da árvore é ***d***, então:

  1. cada nó folha está no nível ***d* - 1** ou no nível ***d***

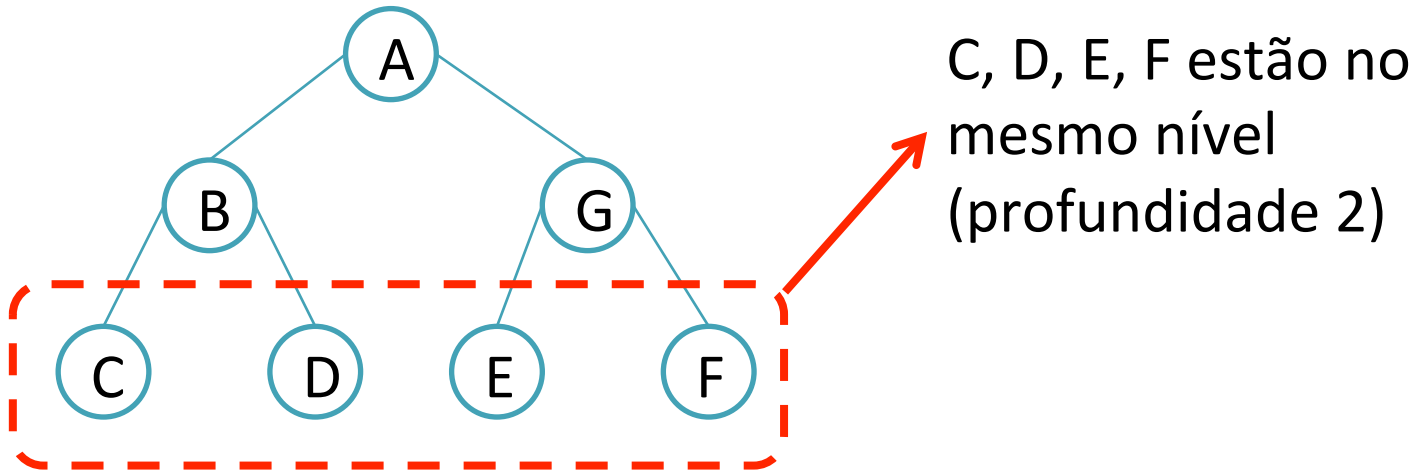
  2. o nível ***d* - 1** está totalmente preenchido

  3. os nós folha no nível ***d*** estão todos mais à esquerda possível



Árvore Binária Completa Cheia

- **Árvore Binária Completa Cheia** (ABCC)
  - É uma Árvore Estritamente Binária
  - Todos os seus nós folha estão no mesmo nível



Árvore Binária Completa Cheia

- Qual é o número total de nós de uma ABCC de profundidade *d*?

Árvore Binária Completa Cheia

- Dada uma **ABCC** e sua profundidade ***d***, calculamos o **número total de nós** na árvore:

Profundidade: **Total (até prof. *d*):**

- *d* = 0 **=>** 1 nó — **1**
- *d* = 1 **=>** 2 nós — **3**
- *d* = 2 **=>** 4 nós — **7**
- ...
- Profundidade ***d*** **=>** 2^d nós — $2^{d+1} - 1$

Árvore Binária Completa Cheia

- Portanto, se o número de nós, ***n***, para uma árvore binária completa cheia de profundidade ***d*** é
  - $n = 2^{d+1} - 1$
- Então, ***n*** nós podem ser distribuídos em uma **árvore binária completa cheia de profundidade** ... ???

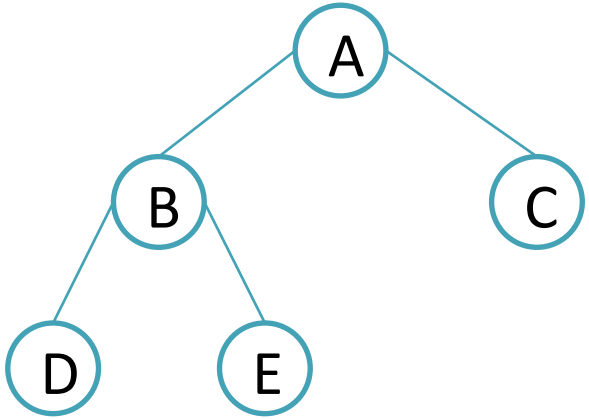
Árvore Binária Completa Cheia

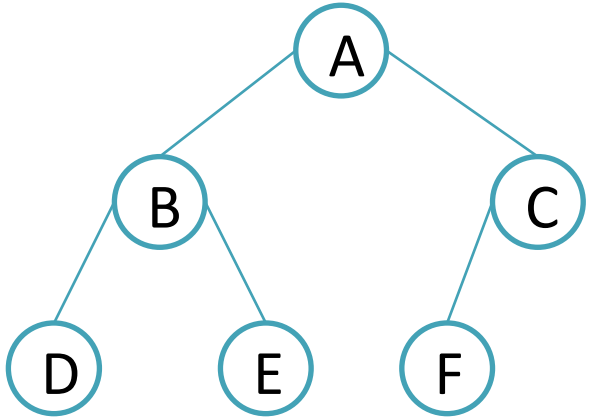
- Uma Árvore Binária Completa Cheia com ***n*** nós terá **profundidade**:
  - $n = 2^{d+1} - 1$
  - $\log_2(n + 1) = \log_2(2^{d+1})$
  - ...

$$d = \log_2(n + 1) - 1$$

Árvore Binária Balanceada

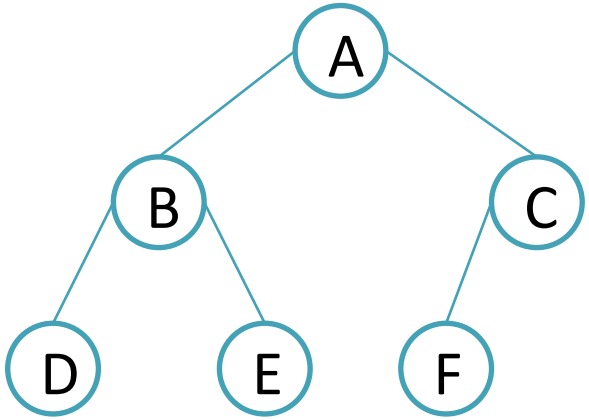
- **Árvore Binária Balanceada**
  - **Para cada nó**, os valores de **altura** de suas duas subárvores diferem de, no máximo, **1**





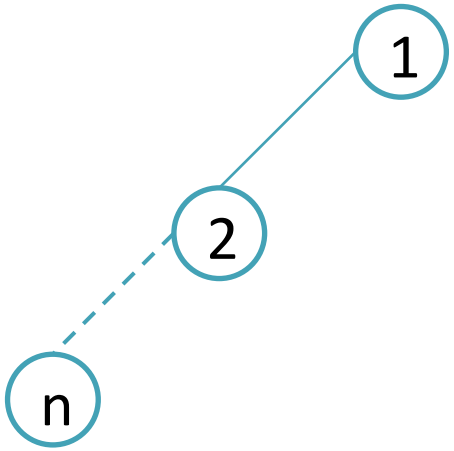
Árvore Binária Perfeitamente Balanceada

- **Árvore Binária Perfeitamente Balanceada**
  - Para cada nó, o **número de nós** de suas sub-árvores esquerda e direita difere em, no máximo, **1**
  - Toda Árvore Binária Perfeitamente Balanceada é Balanceada, mas o inverso não é necessariamente verdade



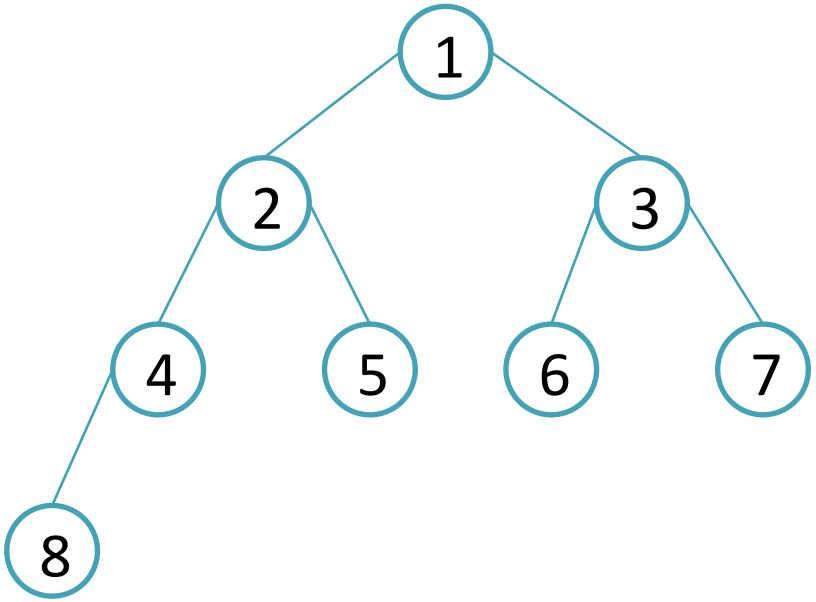
Árvore Binária

- Qual a **altura máxima** de uma AB com *n* nós?
  - Resposta: **n-1**
  - **Árvore Degenerada** => Lista



Árvore Binária

- Qual a **altura mínima** de uma AB com *n* nós?
  - **Resposta:** a mesma de uma **AB Perfeitamente Balanceada** com ***n*** nós



$n = 1$ ------ $h = 0$
$n = 2, 3$ --- $h = 1$
$n = 4...7$ -- $h = 2$
$n = 8...15$ - $h = 3$

$$h_{\min} = \lfloor \log_2 n \rfloor$$

Exercício

- Escreva um algoritmo para uma função recursiva que calcula a altura de uma AB

AB - Percursos

- Percorrer uma AB "visitando" cada nó uma única vez
  - "Visitar" um nó pode ser
    - Mostrar o seu valor
    - Modificar o valor do nó
    - ...

AB - Percursos

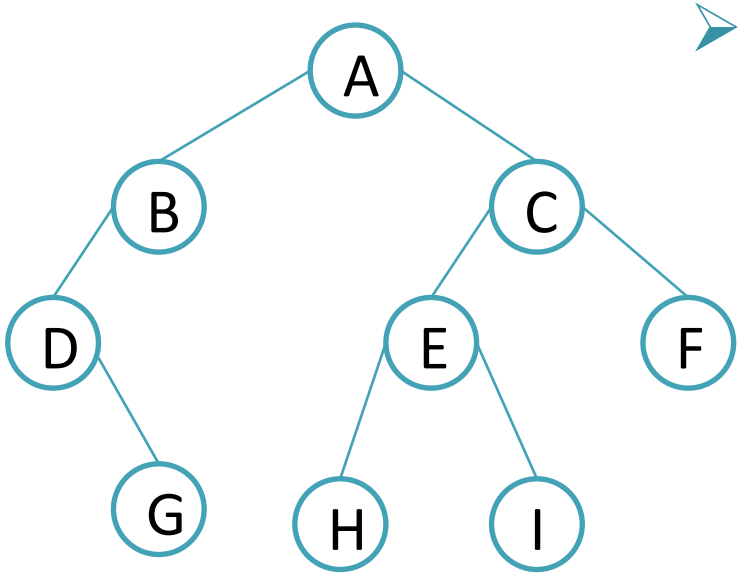
- **PERCURSO** => sequência linear de nós
  - podemos então falar em nó predecessor ou sucessor de outro nó, segundo um dado percurso
- Não existe um percurso único para árvores (binárias ou não)
  - diferentes percursos podem ser realizados, dependendo da aplicação

AB - Percursos em Árvores

- 3 percursos básicos para AB:
  - **pré-ordem** (Pre-order)
  - **em-ordem** (In-order)
  - **pós-ordem** (Post-order)
- A diferença entre eles está, basicamente, na ordem em que os nós são "visitados"

AB - Percurso Pré-Ordem

- ✓ visita a raiz
- ✓ percorre a subárvore à esquerda em **pré-ordem**
- ✓ percorre a subárvore à direita em **pré-ordem**

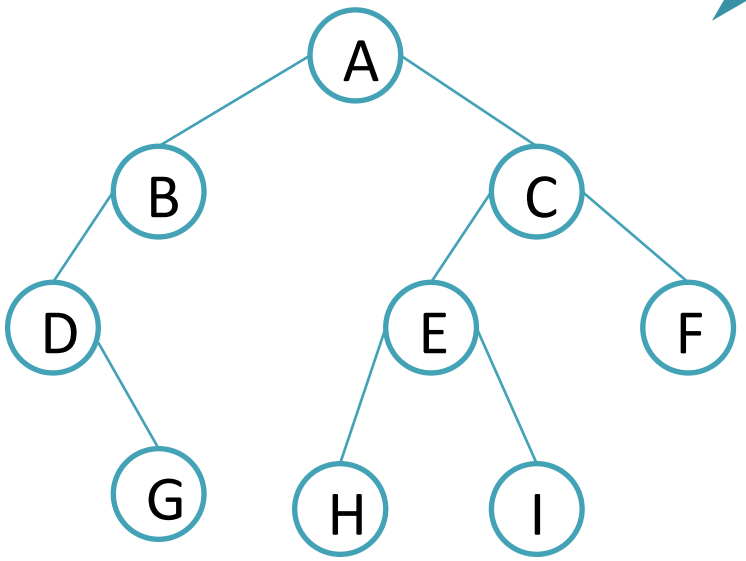


- Resultado:

A B D G C E H I F

AB - Percurso Em-Ordem

- ✓ percorre a subárvore à esquerda **em-ordem**
- ✓ visita a raiz
- ✓ percorre a subárvore à direita **em-ordem**

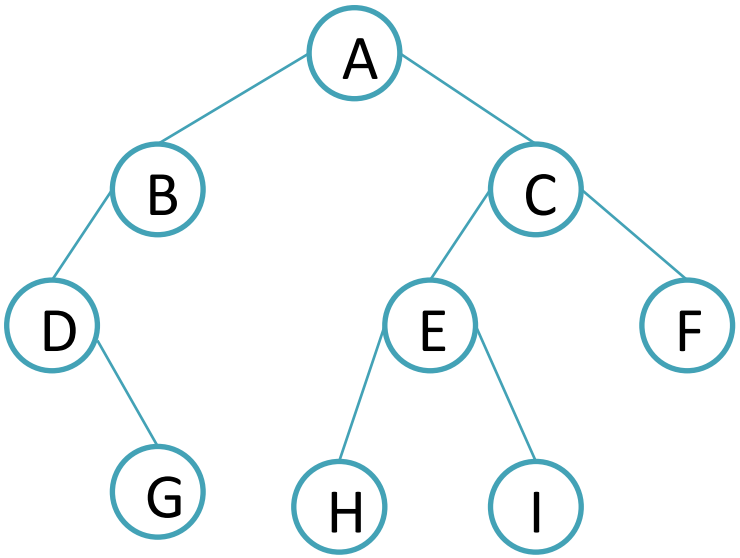


- Resultado:

D G B A H E I C F

AB - Percurso Pós-Ordem

- ✓ percorre e subárvore a esquerda em **pós-ordem**
- ✓ percorre a subárvore a direita em **pós-ordem**
- ✓ visita a raiz

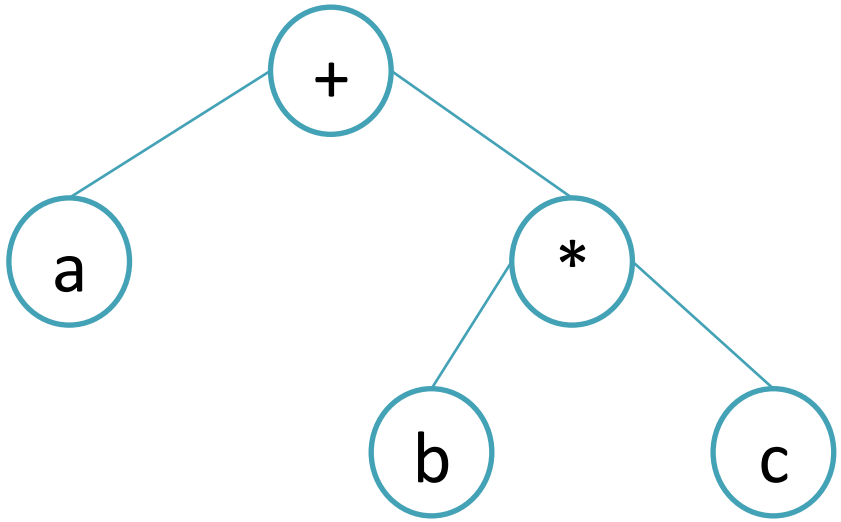


- Resultado:

G D B H I E F C A

AB - Percursos

- Percurso para expressões aritméticas
  - Pré-ordem: **+a*bc**
  - Em-ordem: **a+(b*c)**
  - Pós-ordem: **abc*+**



Próxima Aula...

- Árvore Binária de Busca
- Implementação