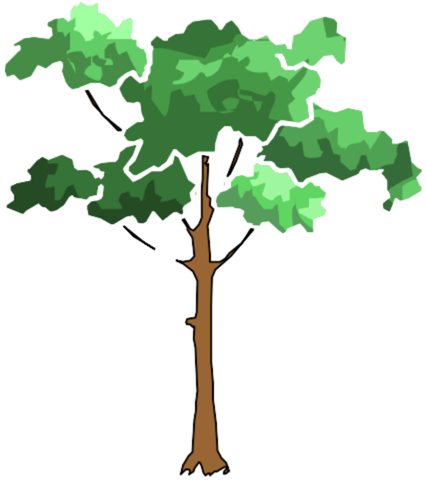
Árvores &
Árvores Binárias



Problema

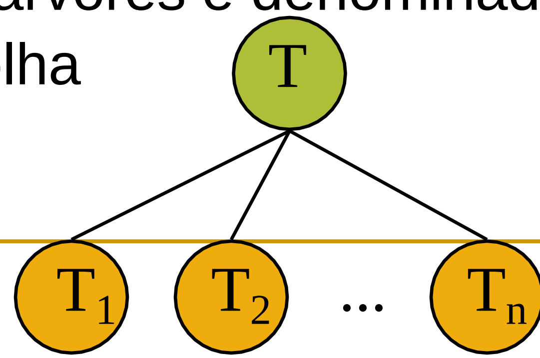
- Implementações do TAD Lista Linear
  - Lista encadeada
    - eficiente para inserção e remoção dinâmica de elementos, mas ineficiente para busca
  - Lista seqüencial (ordenada)
    - Eficiente para busca, mas ineficiente para inserção e remoção de elementos
- Árvores: solução eficiente para inserção, remoção e busca
  - Representação não linear...

Definições

- Árvore T: conjunto finito de elementos, denominados **nós** ou vértices, tais que:
  - Se T = ∅, a árvore é dita vazia; c.c.

    (i) T contém um nó especial, denominado raiz;

    (ii) os demais nós, ou constituem um único conjunto vazio, ou são divididos em $n \geq 1$ conjuntos disjuntos não vazios $(T_1, T_2, \ldots, T_n)$, que são, por sua vez, cada qual uma árvore;
  - $T_1, T_2, \ldots, T_n$ são chamadas sub-árvores de T;
  - Um nó sem sub-árvores é denominado nó-folha, ou simplesmente, folha

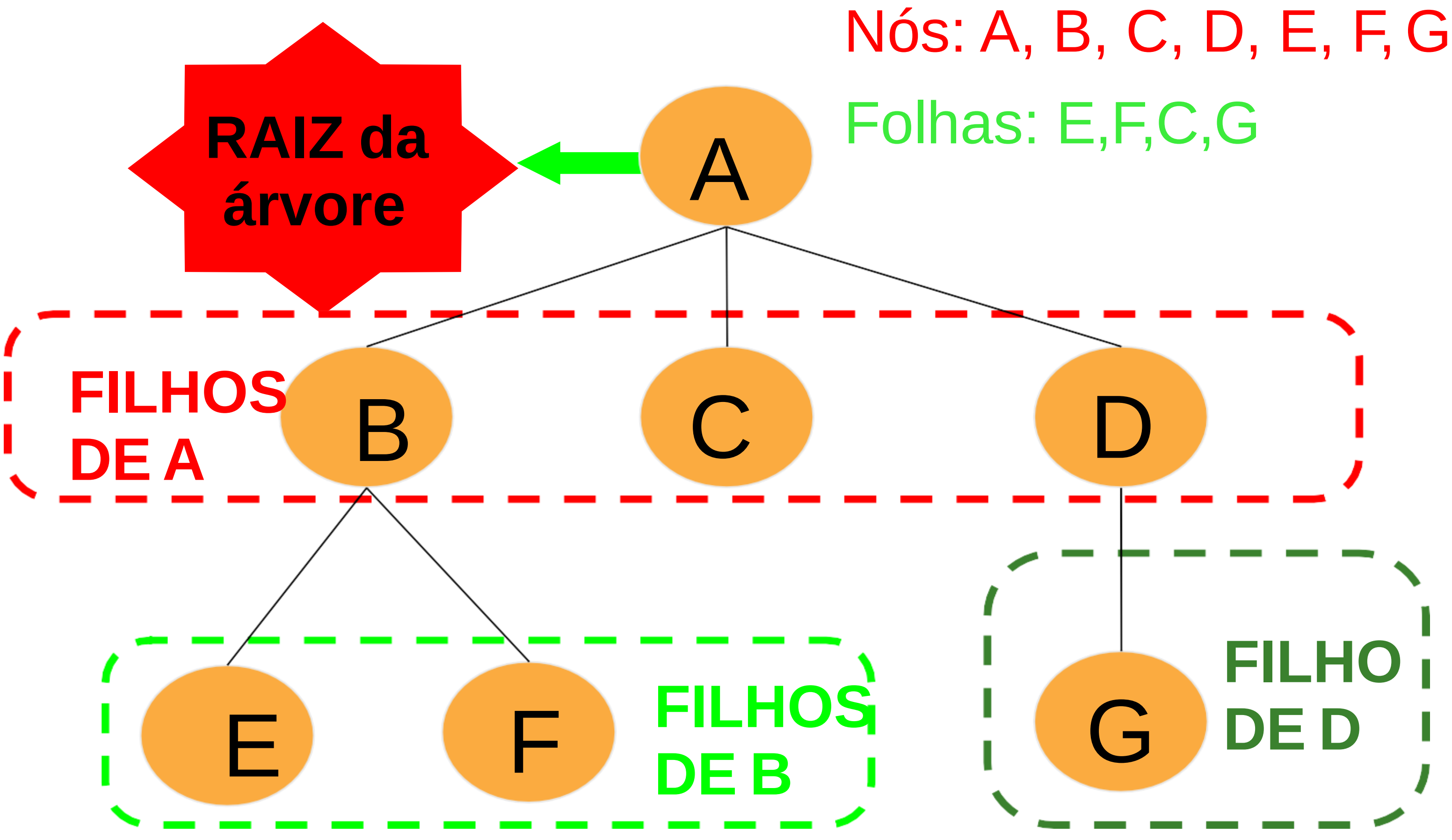


Definições (cont.)

- Árvore: adequada para representar estruturas hierárquicas não lineares, como relações de descendência (pai, filho, irmãos, etc.)

- Se um nó X é raiz de uma árvore, e um nó Y é raiz de uma sub-árvore de X, então X é **PAI** de Y e Y é **FILHO** de X

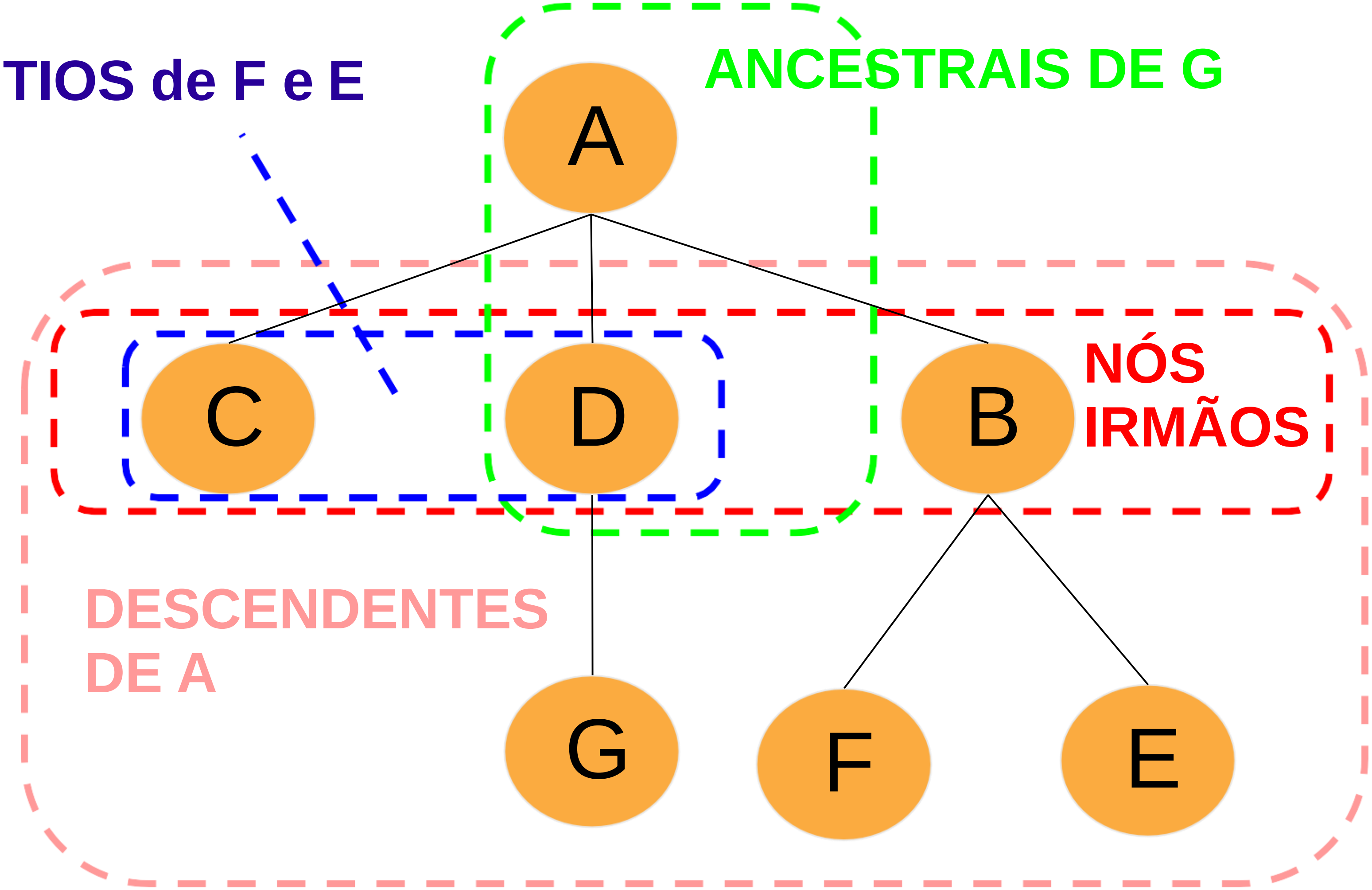
Definições (cont.)



Definições (cont.)

- O nó X é um ANCESTRAL do nó Y (e Y é DESCENDENTE de X) se X é o PAI de Y, ou se X é PAI de algum ANCESTRAL de Y
- Dois nós são IRMÃOS se são filhos do mesmo pai
- Se os nós Y_1, Y_2, ...Y_j são irmãos, e o nó Z é filho de Y_1, então Y_2,...Y_j são TIOs de Z

Definições (cont.)



Conceitos

- O **NÍVEL** de um nó **X** é definido como:
  - O nível do nó raiz é 0
  - O nível de um nó não-raiz é dado por (nível de seu nó PAI + 1)
- Os nós de maior nível são também nós-folha.

Conceitos (cont.)

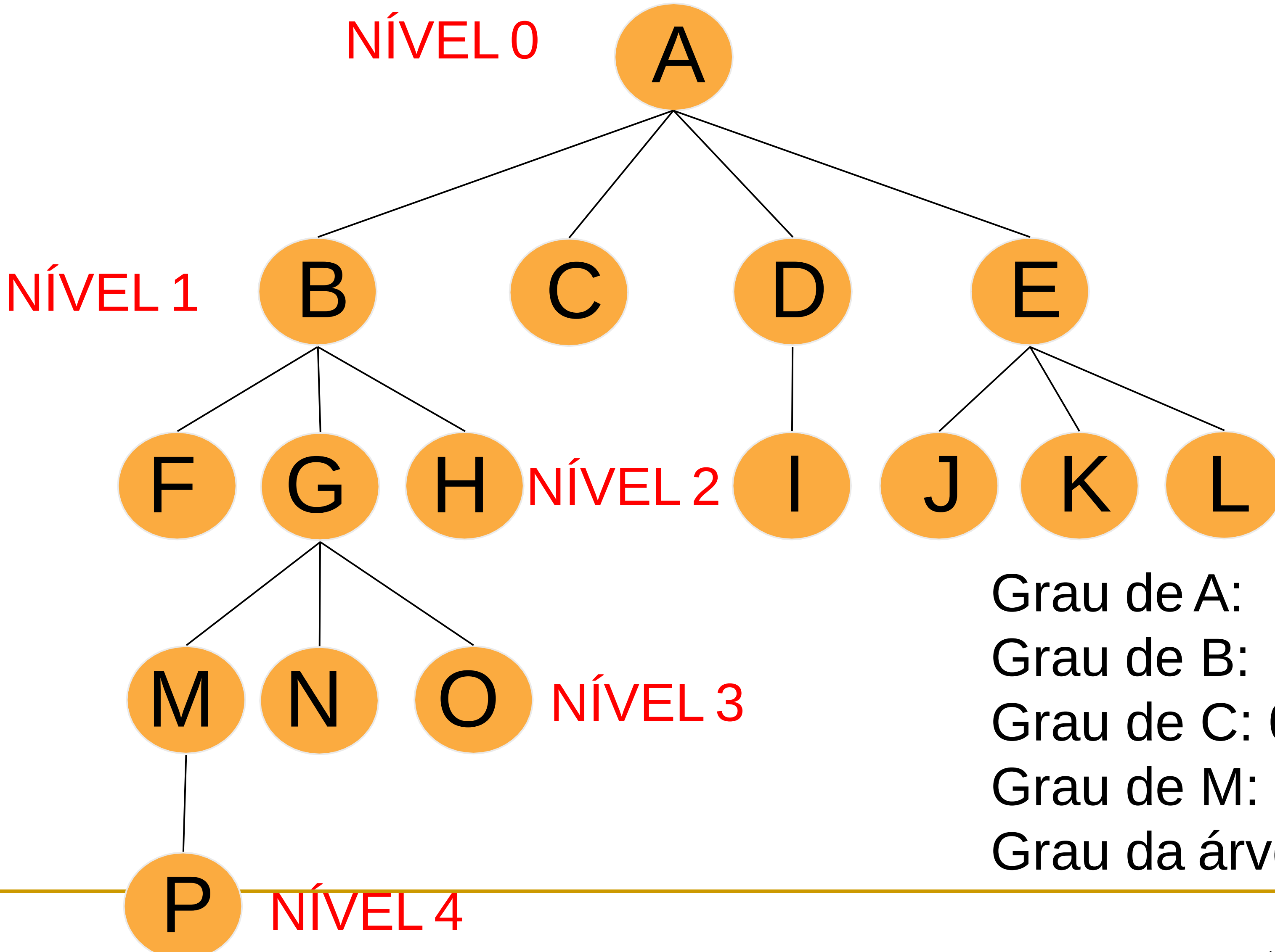
- O **GRAU de um nó X** pertencente a uma árvore é igual ao número de filhos do nó X

- O **GRAU de uma árvore T** é o maior entre os graus de todos os seus nós

Conceitos (cont.)

- A **ALTURA** de um nó **X** é a distância dele até os nó folha, ou seja, de cima para baixo.
- A **PROFUNDIDADE** de um nó **X** é a distância dele até a raiz da árvore, ou seja, de baixo para cima.

Conceitos (cont.)



Grau de A: 4

Grau de B: 3

Grau de C: 0

Grau de M: 1

Grau da árvore:4

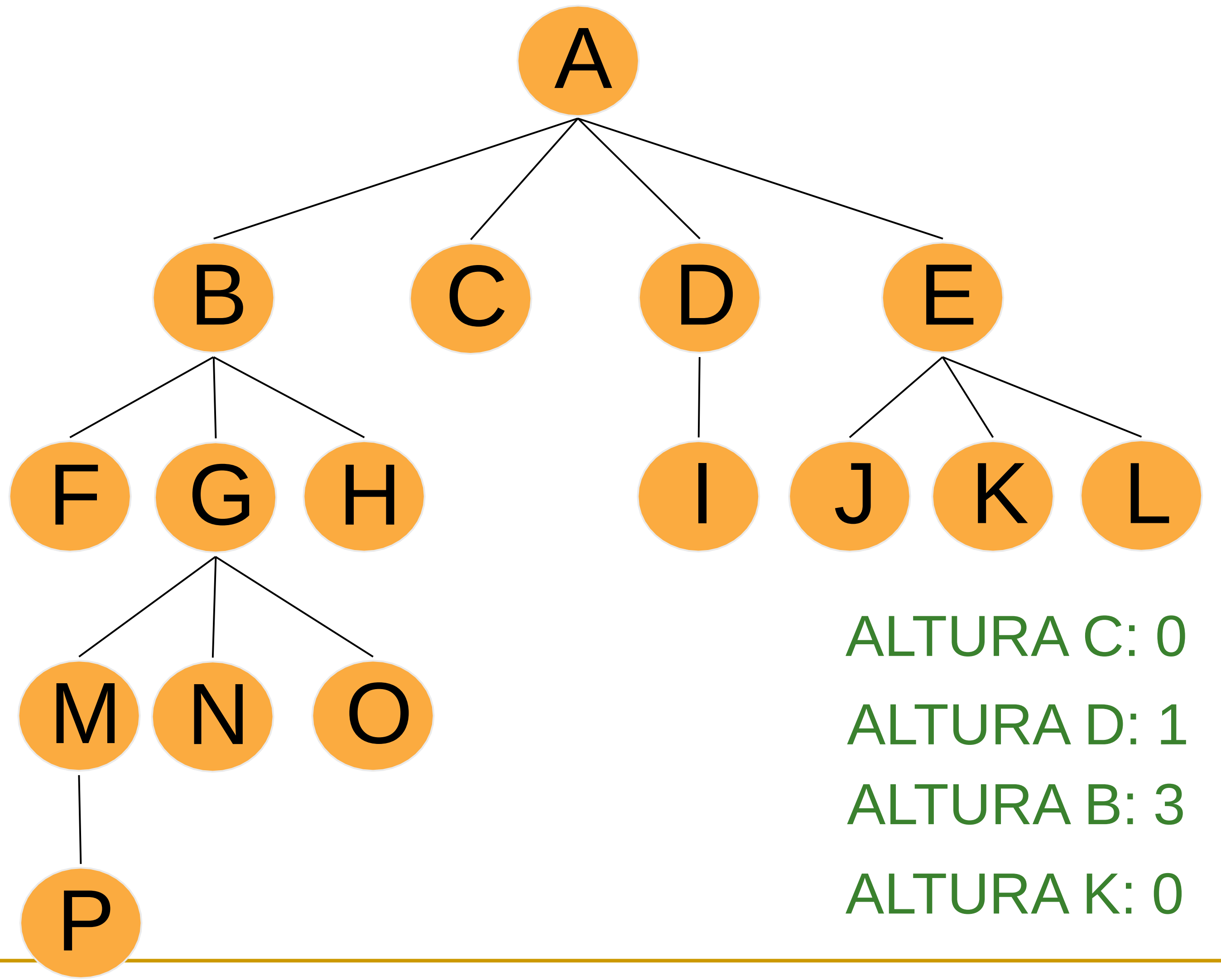
Conceitos (cont.)

- Uma sequência de nós distintos v_1, ….v_k tal que cada nó v_{i+1} é filho de v_i é denominada um **CAMINHO** na árvore (diz-se que v_i alcança v_k).

- O número de arestas de um caminho define o **COMPRIMENTO** DO CAMINHO.

- Denota-se a altura de uma árvore com raiz **X** por **h(X)**, e a altura de uma sub-árvore com raiz **y** por **h(y)**

Conceitos (cont.)



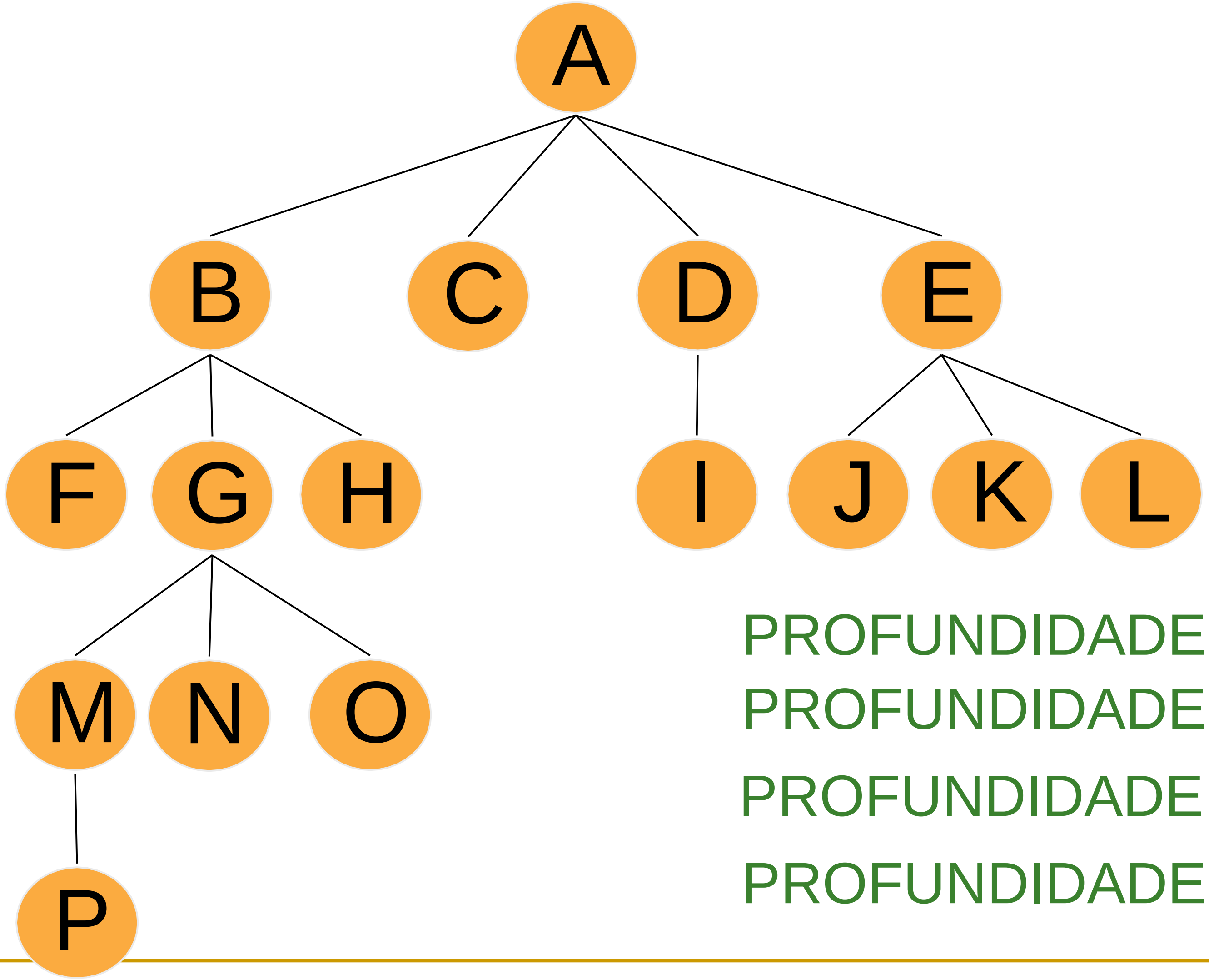
ALTURA C: 0

ALTURA D: 1

ALTURA B: 3

ALTURA K: 0

Conceitos (cont.)



PROFUNDIDADE C: 1

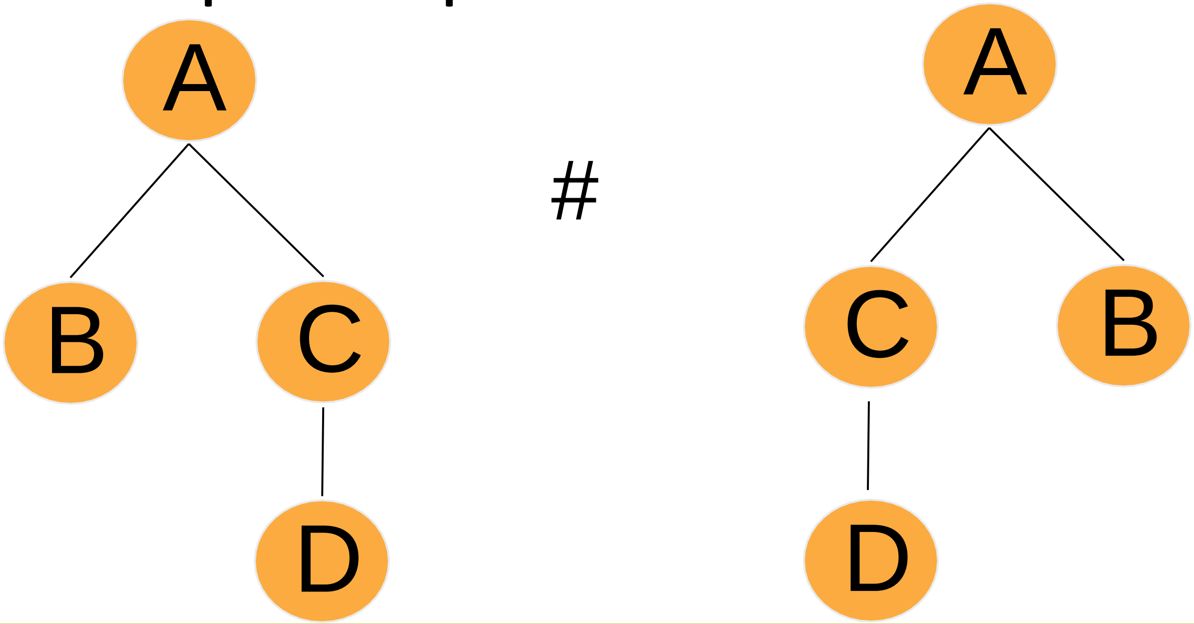
PROFUNDIDADE I: 2

PROFUNDIDADE O: 3

PROFUNDIDADE F: 2

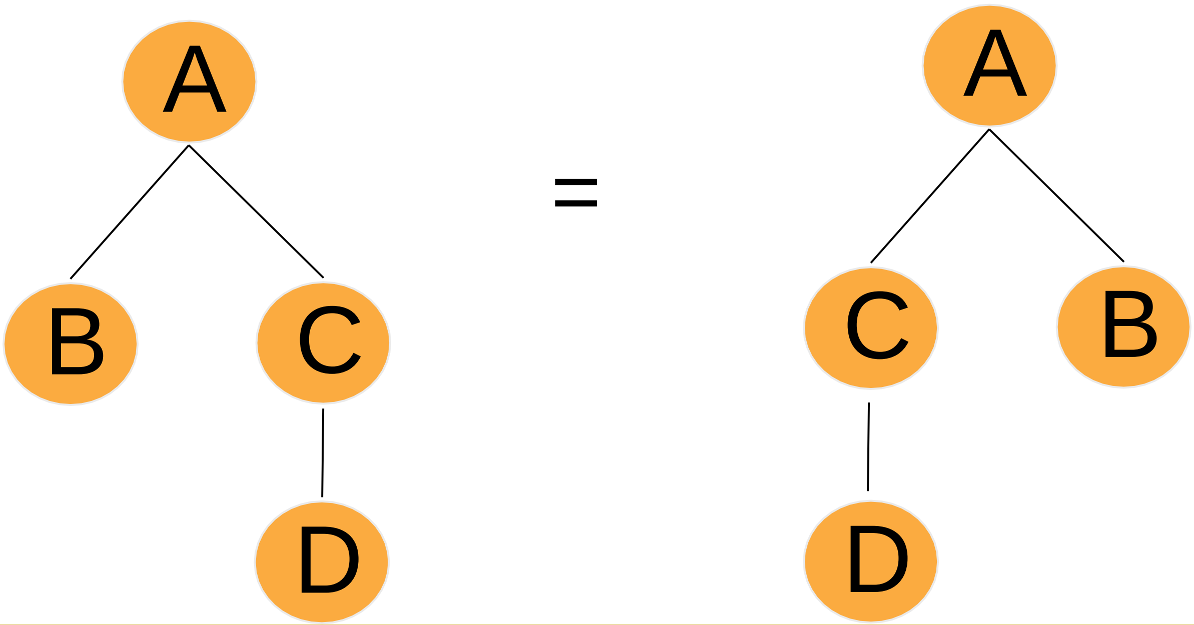
Conceitos (cont.)

- Uma árvore é **ORDENADA** se considerarmos o conjunto de sub-árvores T_1, T_2, …T_n como um conjunto ordenado. Geralmente a ordenação se dá da esquerda para a direita



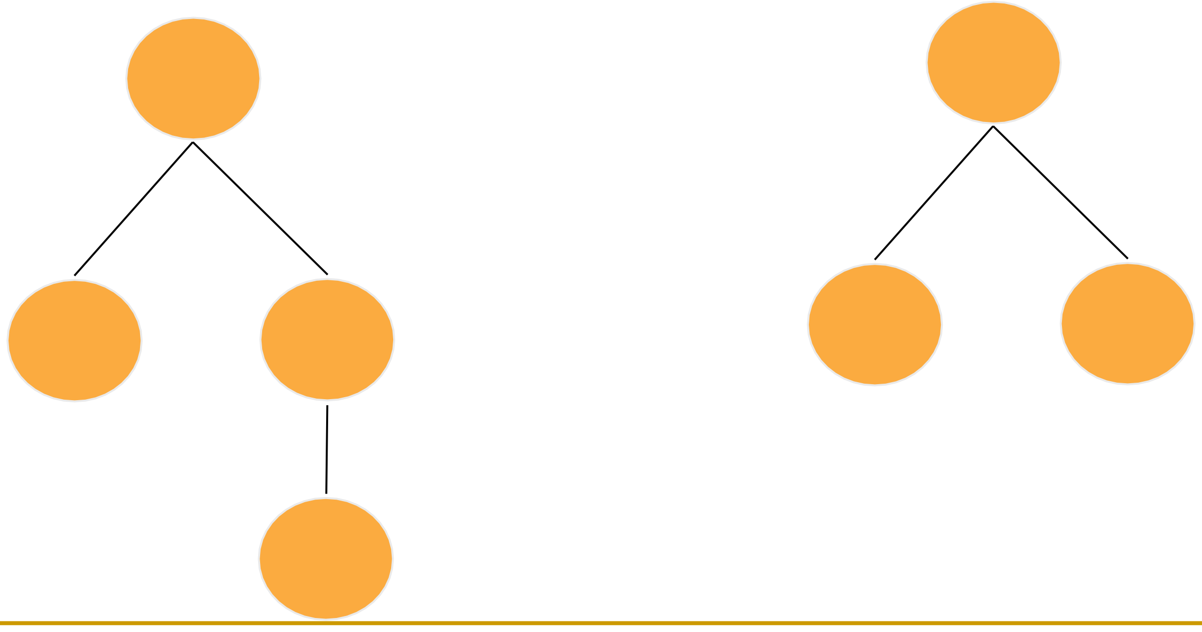
Conceitos (cont.)

Quando a ordem das sub-árvores não é relevante, dizemos que a árvore é **orientada**, uma vez que apenas a orientação dos nós é importante



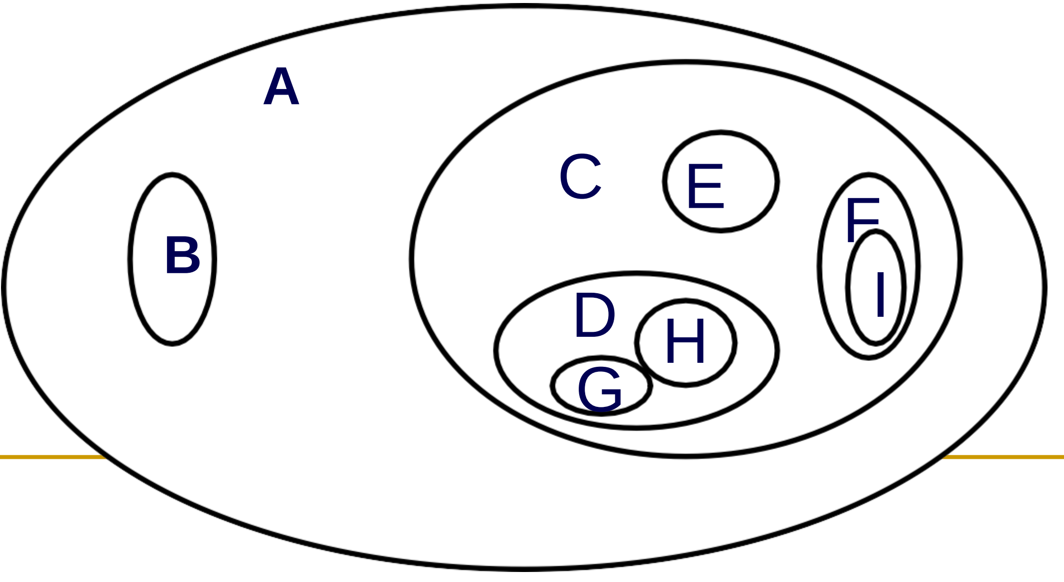
Conceitos (cont.)

- Uma **FLORESTA** é um conjunto de 0 ou mais árvores distintas



Outras Representações Gráficas

- Representação por parênteses aninhados
  - (A (B) (C (D (G) (H)) (E) (F (I))))
  - ou seja, uma lista generalizada!!
- Representação por Diagramas de Venn



Árvore Binárias (AB)

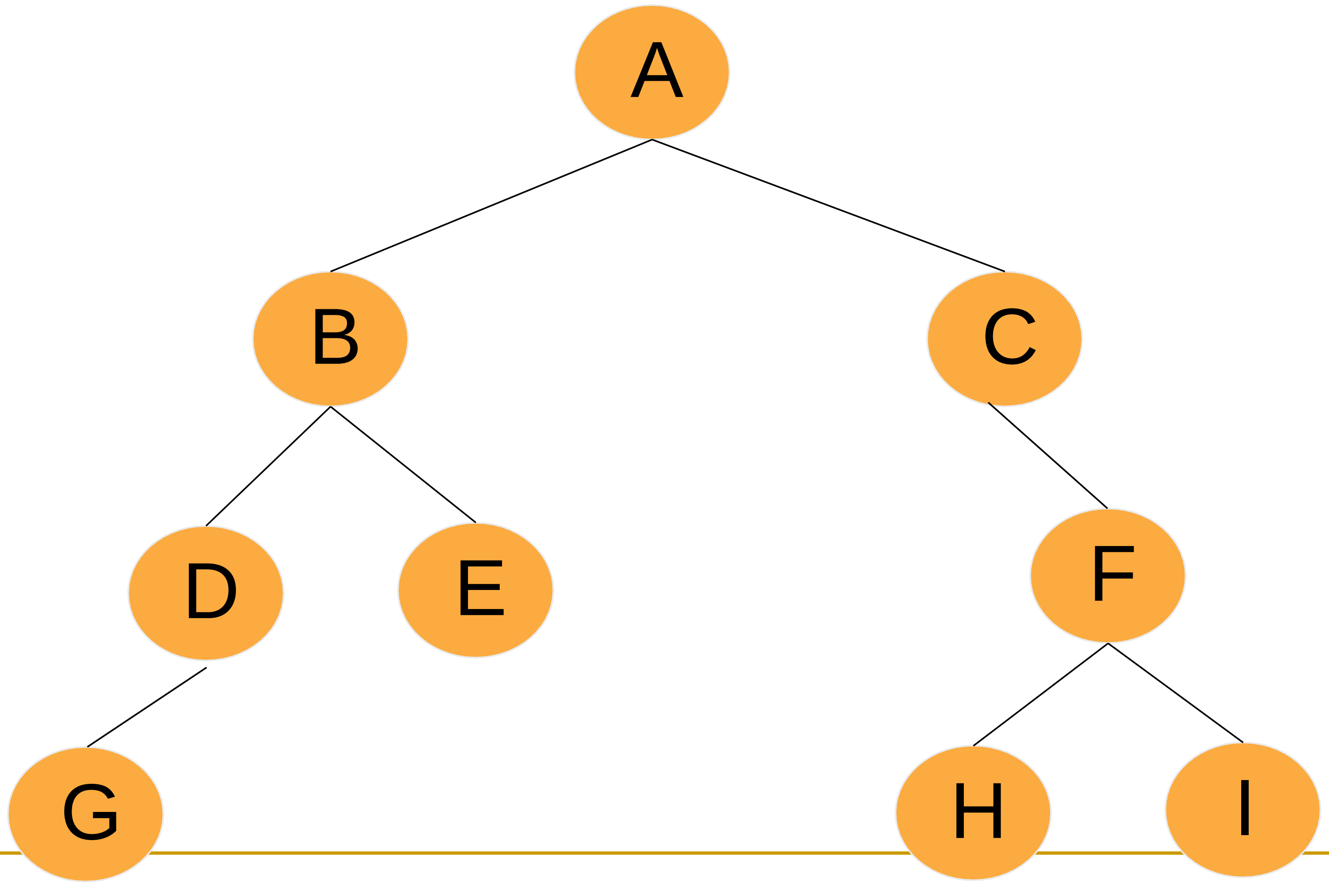
- Uma Árvore Binária (AB) T é um conjunto finito de elementos, denominados nós ou vértices, tal que:
  - (i) Se T = ∅, a árvore é dita vazia, ou
  - (ii) T contém um nó especial, chamado raiz de T, e os demais nós podem ser subdivididos em SOMENTE dois sub-conjuntos distintos T_E e T_D, os quais também são árvores binárias.
  - T_E e T_D são denominados sub-árvore esquerda e sub-árvore direita de T, respectivamente

Árvore Binárias (AB) (cont.)

- A raiz da sub-árvore esquerda (ou direita) de um nó **v**, se existir, é denominada filho esquerdo (ou direito) de **v**. Pela natureza da árvore binária, o filho esquerdo pode existir sem o direito, e vice-versa

- A ordem de inserção dos nós na árvore interfere no formato e em toda a estrutura da AB.

Árvores Binárias (AB)



Árvores Binárias (AB) (exemplo)

Duas ABs são SIMILARES se as estruturas das sub-árvores da esquerda e da direita são similares (independente dos valores nos mesmos).

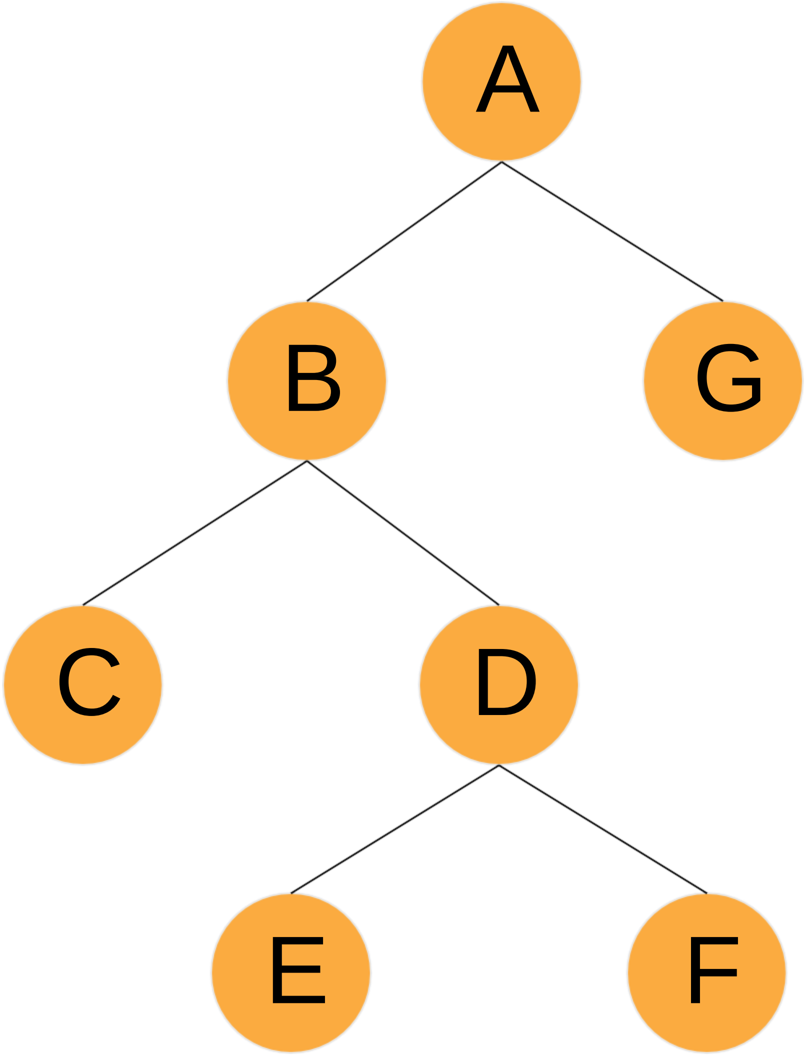
Árvores Binárias (AB) (exemplo)

Duas ABs são IGUAIS se ambas são vazias ou então se armazenam **valores** iguais em suas raizes, suas sub-árvores esquerdas são iguais, e suas sub-árvores direitas também são iguais.

Árvore Estritamente Binária

- Uma **Árvore Binária Simples**, pode ter 1 ou 2 filhos.

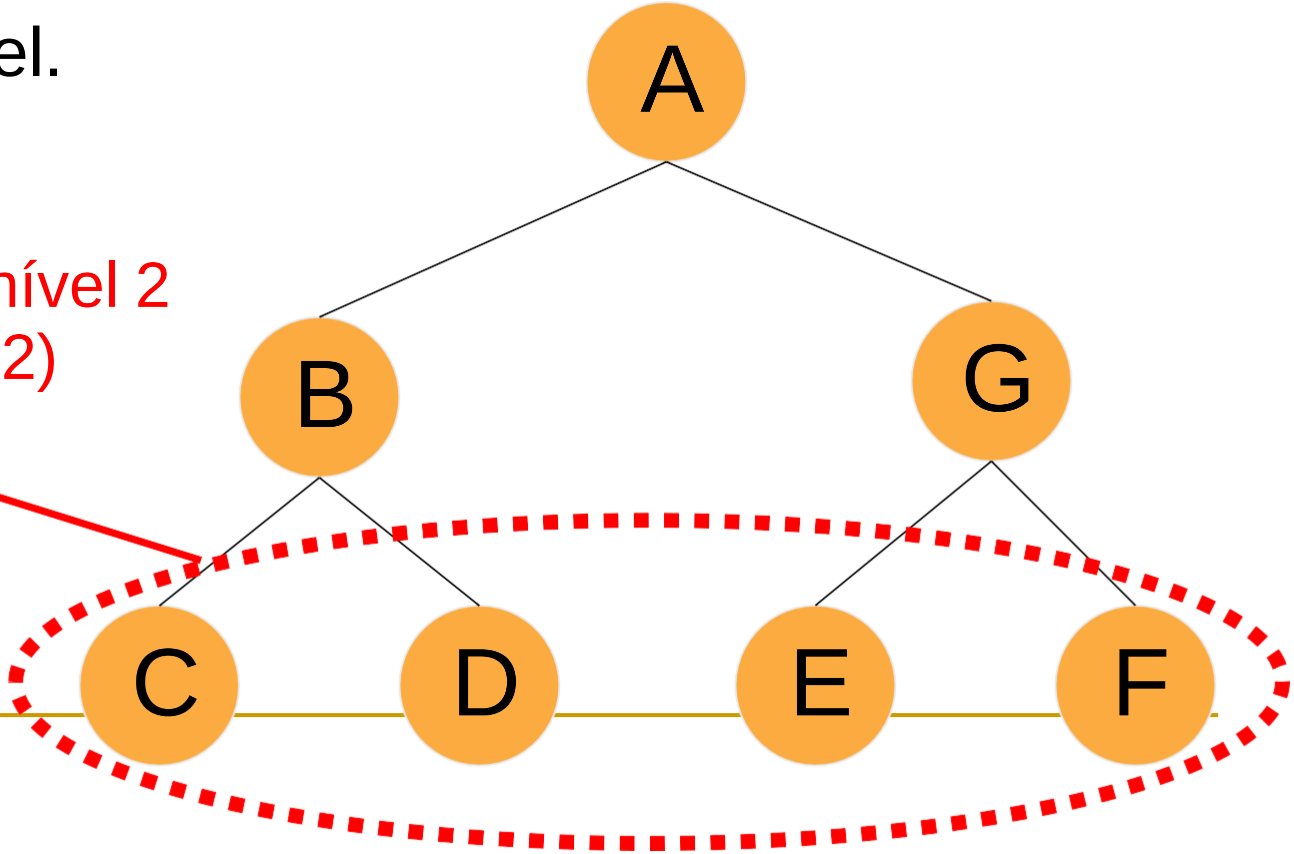
- Uma **Árvore Estritamente Binária** tem nós que têm ou 0 (nenhum) ou dois filhos



Árvore Binária Completa

- Árvore Binária Completa (ABC)
  - é estritamente binária; e
  - todos os seus nós-folha estão no mesmo nível.

C,D,E,F estão no nível 2
(altura da árvore = 2)



Árvore Binária Completa (cont.)

- Dada uma ABC e sua altura, pode-se calcular o número total de nós na árvore.
  - p.ex., uma ABC com profundidade 3 tem 7 nós
    - Nível 0 tem 1 nó
    - Nível 1 tem 2 nós
    - Nível 2 tem 4 nós
    - No. Total de nós = 1 + 2 + 4 = 7
  - Verifique que: se uma ABC tem altura h, então o número de nós da árvore é dado por:

$$\boxed{N = 2^h - 1}$$

Número de nós por nível

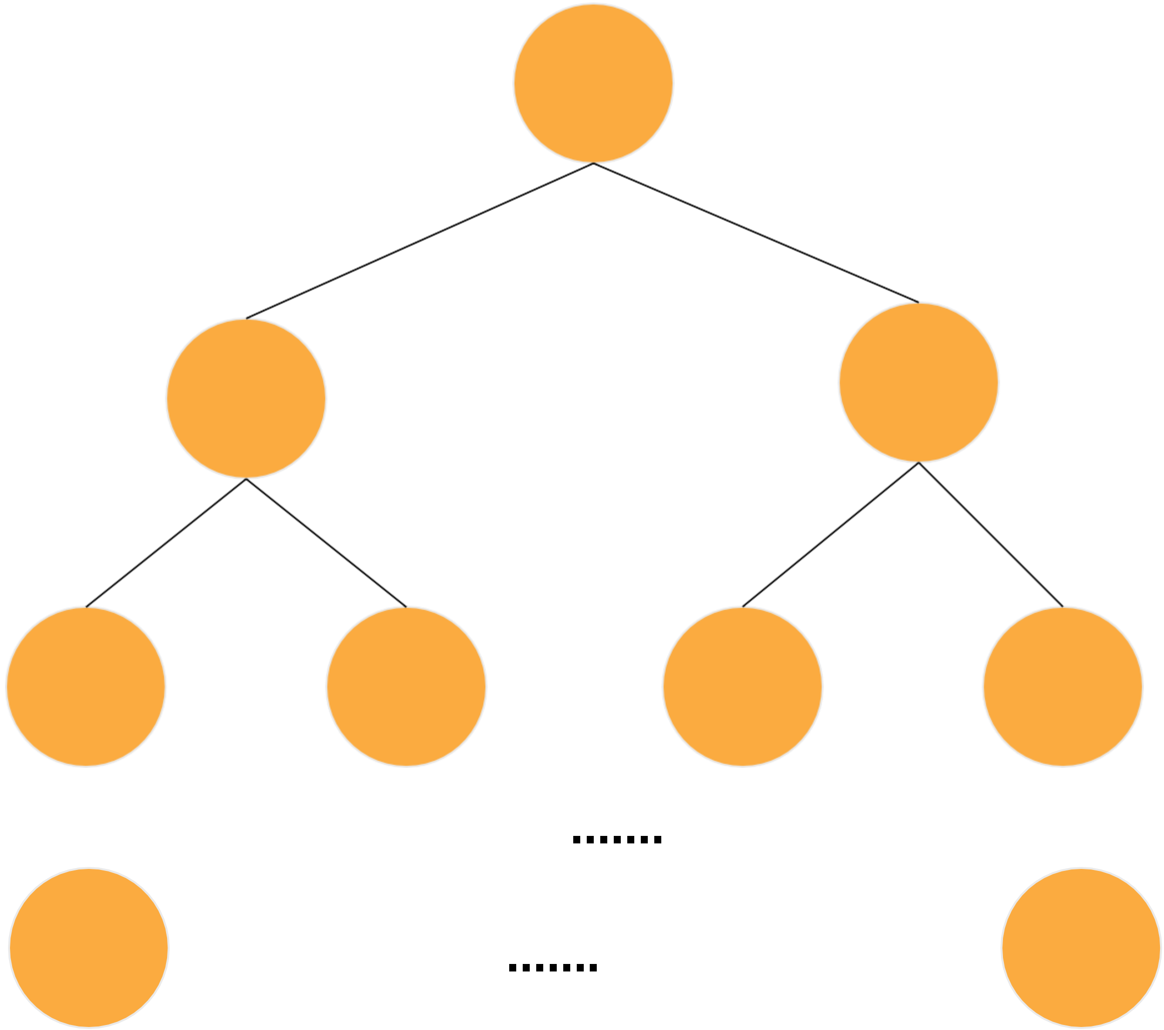
Nível

1 — $1 = 2^0$

2 — $2 = 2^1$

3 — $4 = 2^2$

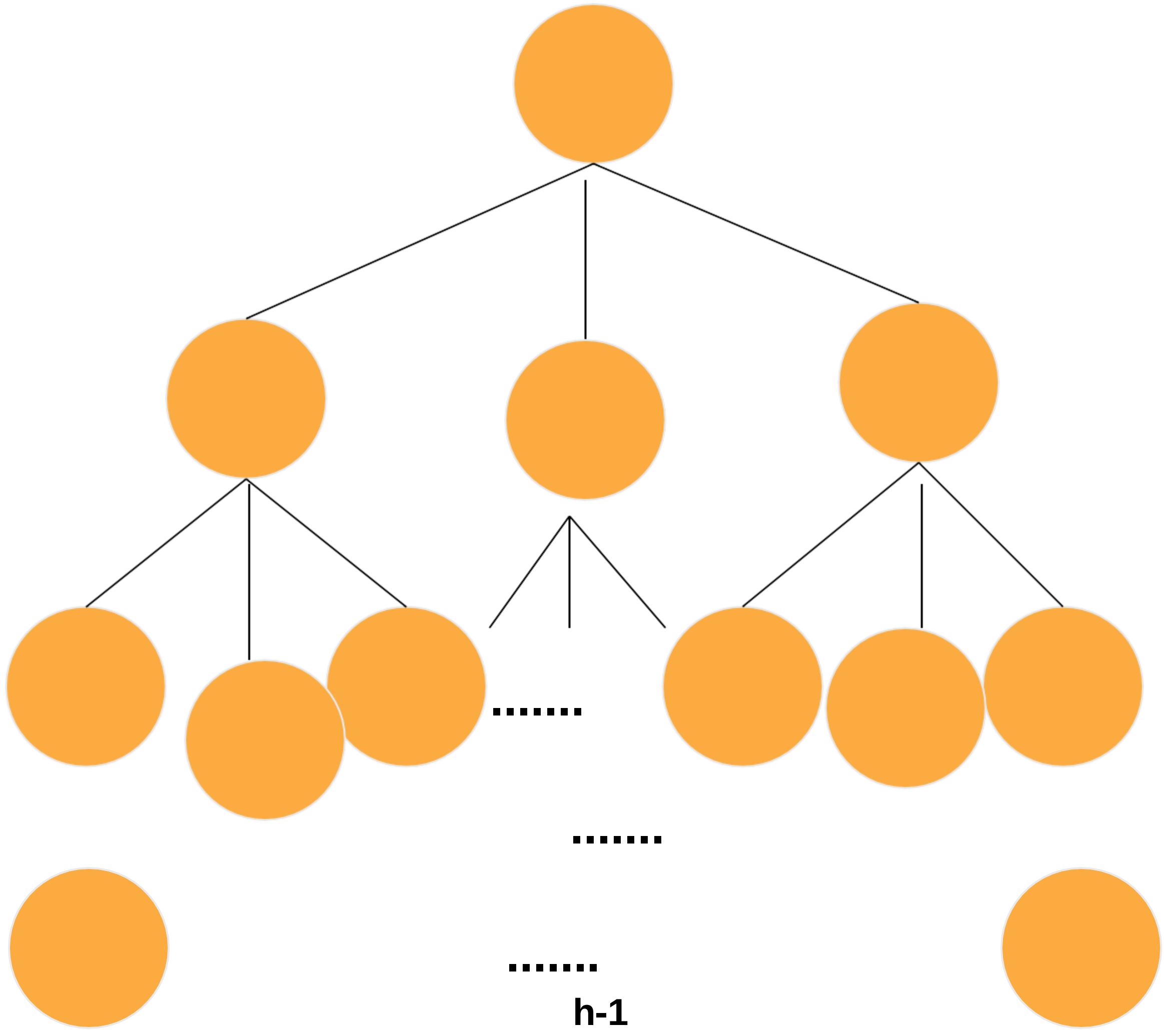
h — 2^{h-1}



$$\therefore N = \sum_{i=0}^{h-1} 2^i = 2^{h-1}$$

Generalize para Árvore Completa de grau **d** e altura **h**

Nível	Número de nós por nível
1	1
2	d
3	d^2
h	d^{h-1}



$$\therefore \; N(h) = \sum_{i=0}^{h-1} d^i = \frac{d^h - 1}{d - 1}$$

Inversamente:

- Se **N** é o número de nós de uma Árvore Binária Completa, de grau **d**, qual é a altura **h** da árvore?

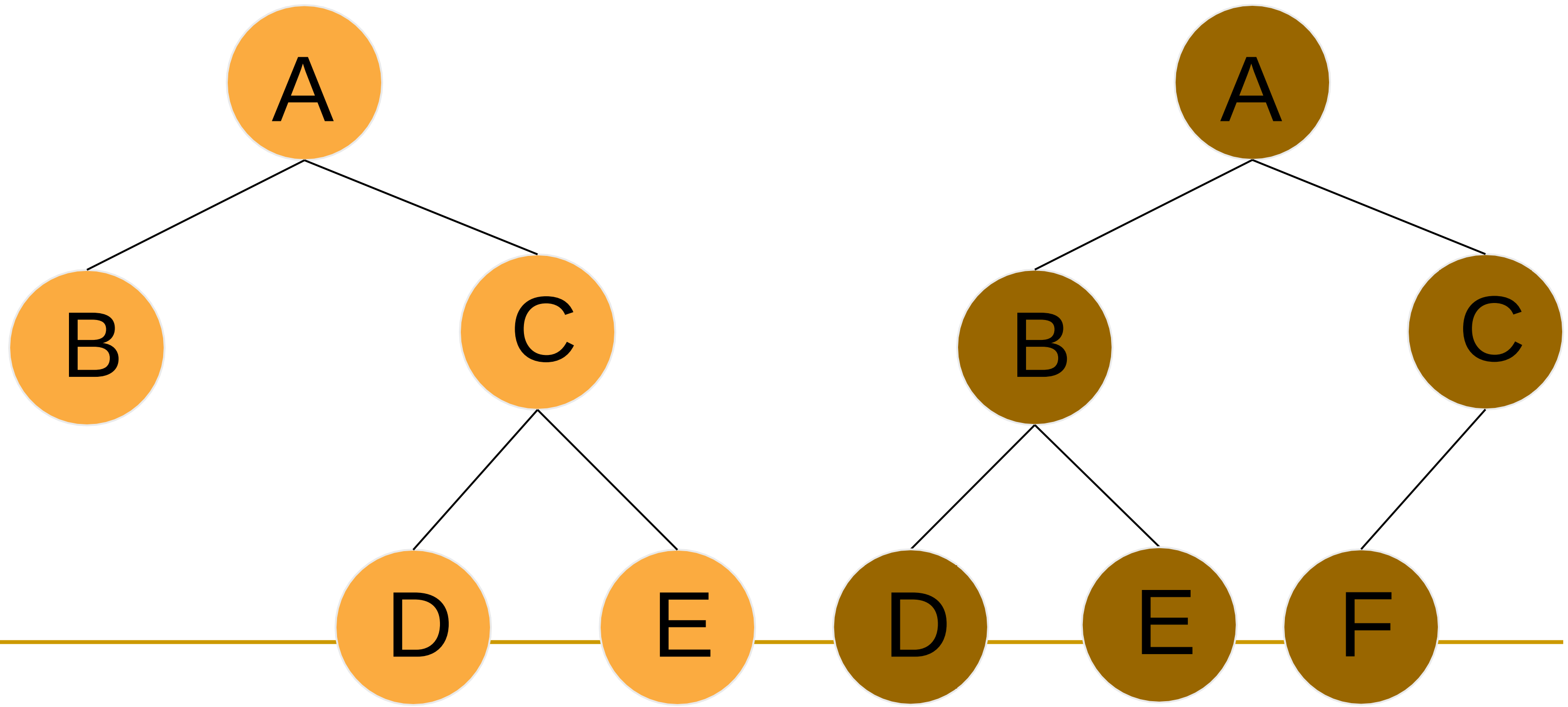
$$\mathbf{N = \frac{d^h - 1}{d - 1}}$$

$$\mathbf{h = \log_d(N.d - N + 1)}$$

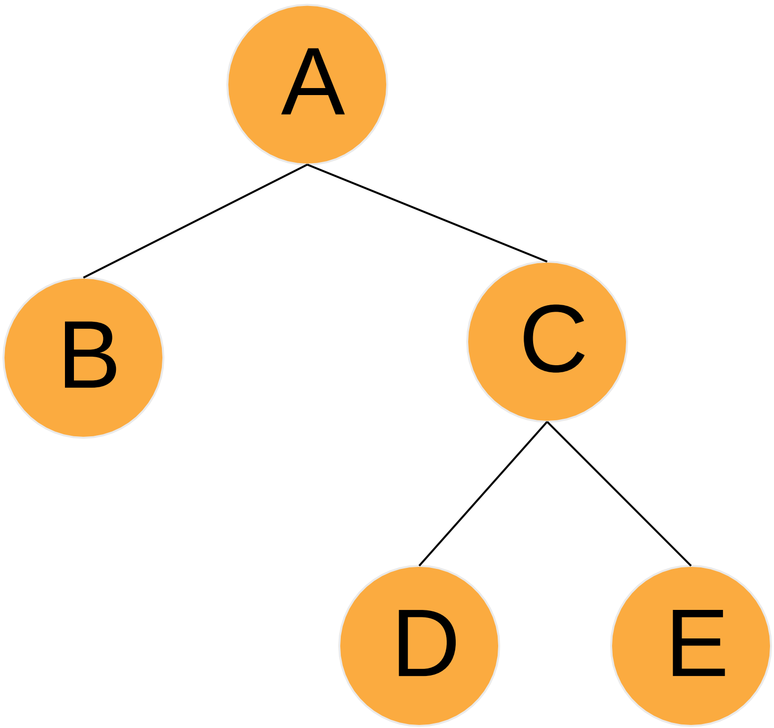
para d=2: $\mathbf{h = \log_2(N + 1)}$

Árvore Binária Balanceada

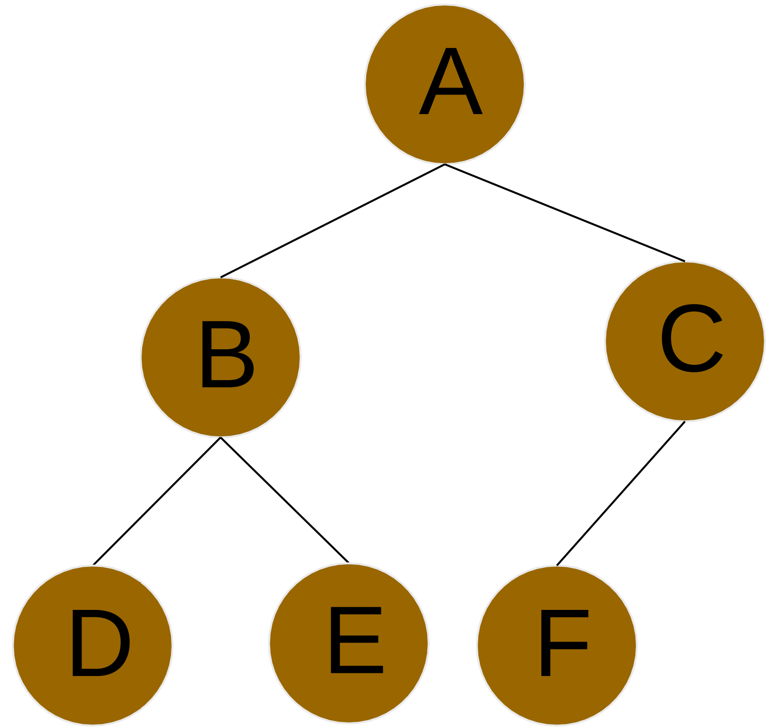
- Árvore Binária Balanceada
  - para cada nó, a diferença das alturas de suas sub-árvores (E e D) tem que ser no máximo 1, em módulo.



Exemplo



Árvore Balanceada

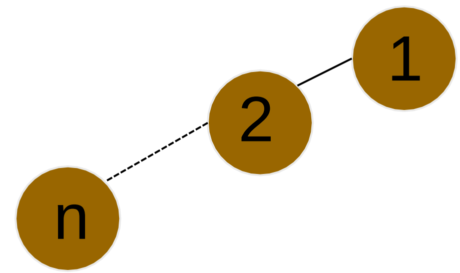


Árvore Perfeitamente Balanceada

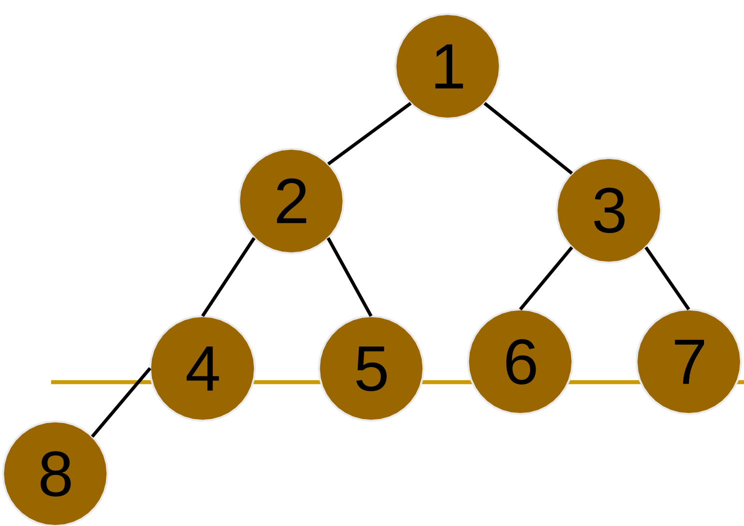
6 nós: $h_{min} = 3$

Questões

- Qual a altura **máxima** de uma AB com **n** nós?
  - Resposta: n
  - Árvore degenerada ≡ Lista



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



N=1; h=1

N=2,3; h=2

N=4..7; h=3

N=8..15; h=4

$$h_{min} = \lfloor \log_2 N \rfloor + 1$$

(maior inteiro ≤ $\log_2 N$) + 1

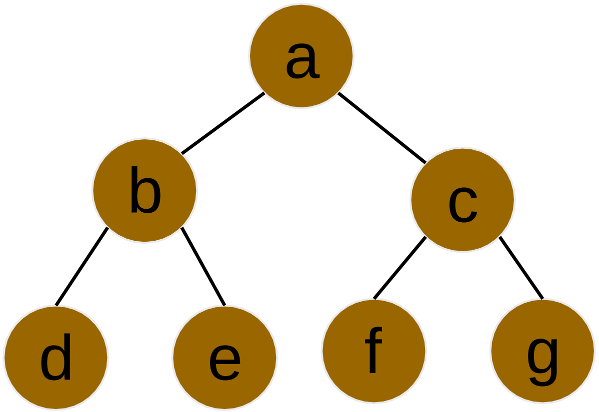
ou

$$h_{min} = \lceil \log_2(N+1) \rceil$$

menor inteiro ≥ $\log_2(N+1)$

Implementação de AB Completa (alocação estática, seqüencial)

- Armazenar os nós, por nível, em um *array*

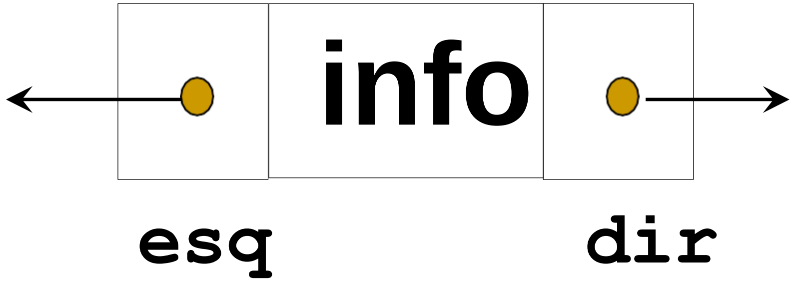


1	2	3	4	5	6	7		...
a	**b**	**c**	**d**	**e**	**f**	**g**		**...**

- Se um nó está na posição i, seus filhos diretos estão nas posições 2i e 2i+1
  - Vantagem: espaço só p/ armazenar conteúdo; ligações implícitas
  - Desvantagem: espaços vagos se árvore não é completa por níveis, ou se sofrer eliminação.

Implementação de AB (dinâmica)

Para qualquer árvore, cada nó é do tipo

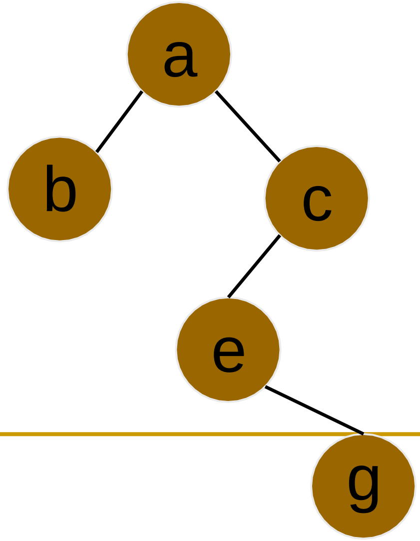


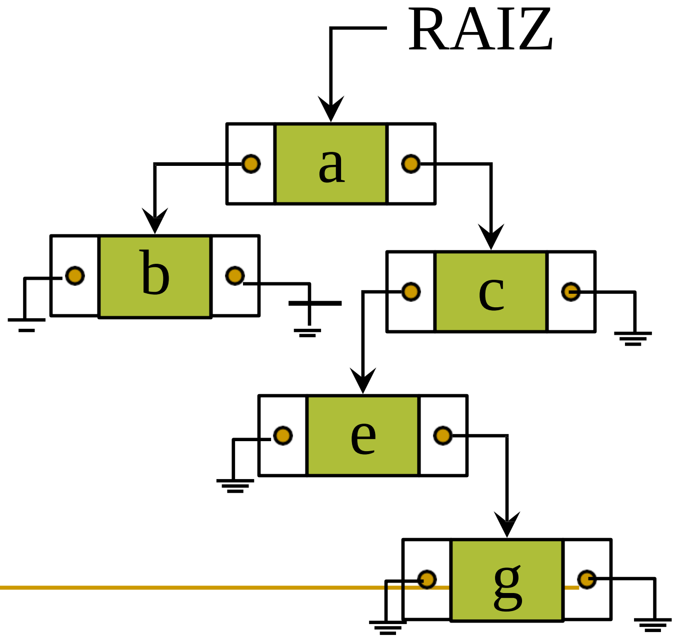

```
typedef struct no *pno;

typedef struct no{
    tipo_elem info;
    pno esq;
    pno dir;
}no;

typedef pno tree;

tree raiz;
```





AB - Percursos

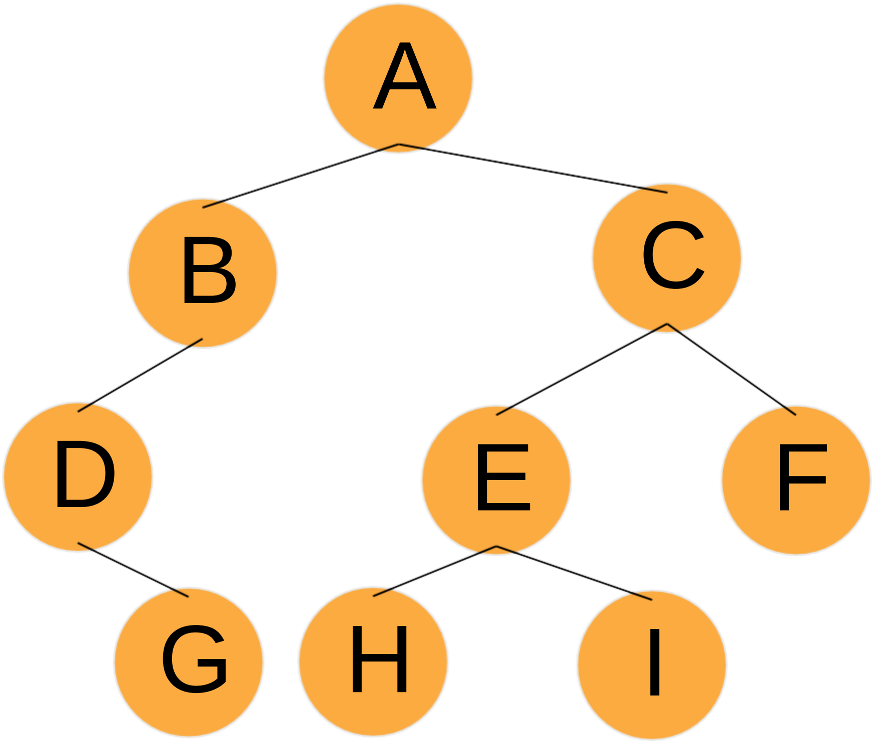
- Objetivo: Percorrer uma AB "visitando" cada nó uma única vez. Um percurso gera uma seqüência linear de nós, e podemos então falar de nó predecessor ou sucessor de um nó, segundo um dado percurso.
- Não existe um percurso único para árvores (binárias ou não): diferentes percursos podem ser realizados, dependendo da aplicação.
- Utilização: imprimir uma árvore, atualizar um campo de cada nó, procurar um item, etc.

AB – Percursos em Árvores

- 3 percursos básicos para AB's:
  - pré-ordem (Pre-order)
  - em-ordem (In-order)
  - pós-ordem (Post-order)
- A diferença entre eles está, basicamente, na ordem em que cada nó é alcançado pelo percurso
  - “Visitar” um nó pode ser:
    - Mostrar (imprimir) o seu valor;
    - Modificar o valor do nó;
    - …

AB - Percurso Pré-Ordem (C, E, D)

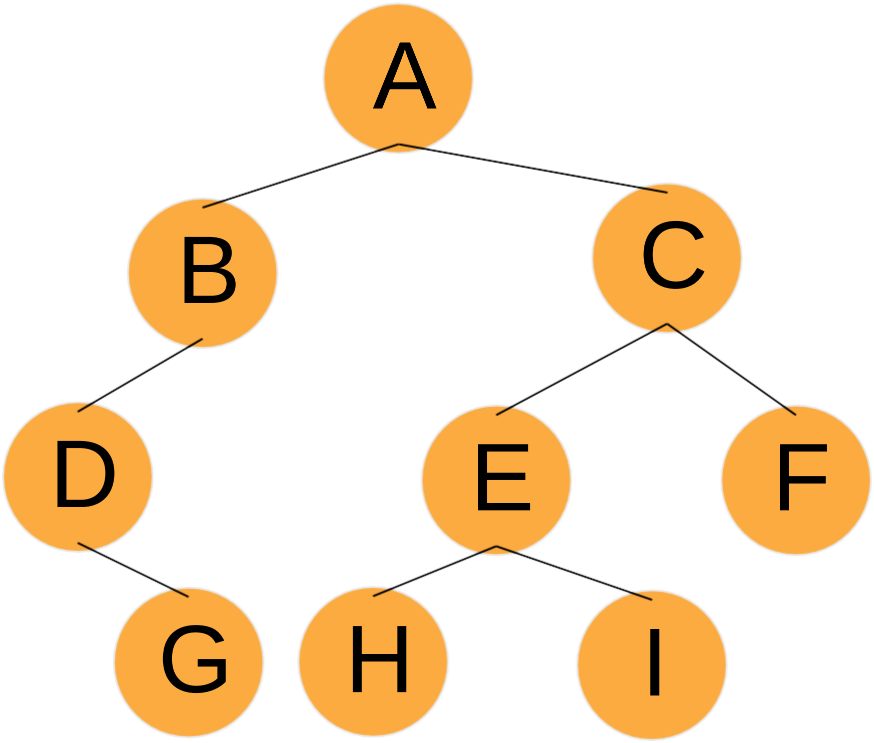
```
void pre_ordem(tree raiz){
    if (raiz != NULL){
        visita(raiz);
        pre_ordem(raiz->esq);
        pre_ordem(raiz->dir);
    }
}
```



Resultado: ABDGCEHIF

AB - Percurso Em-Ordem (E, C, D)

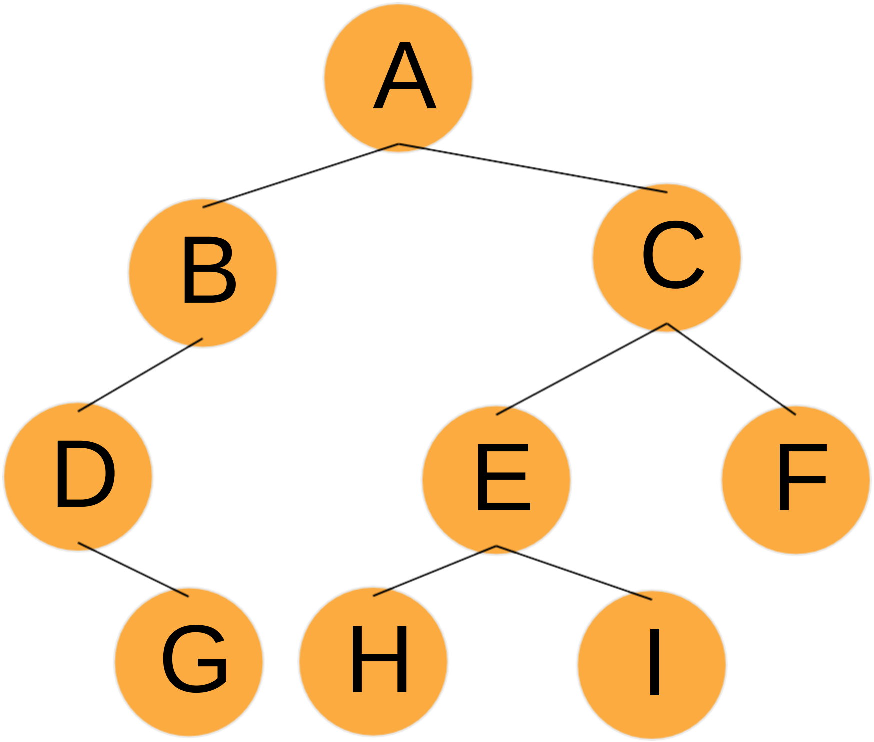
```
void in_ordem(tree raiz){
    if (raiz != NULL){
        in_ordem(raiz->esq);
        visita(raiz);
        in_ordem(raiz->dir);
    }
}
```



Resultado: DGBAHEICF

AB - Percurso Pós-Ordem (E, D, C)

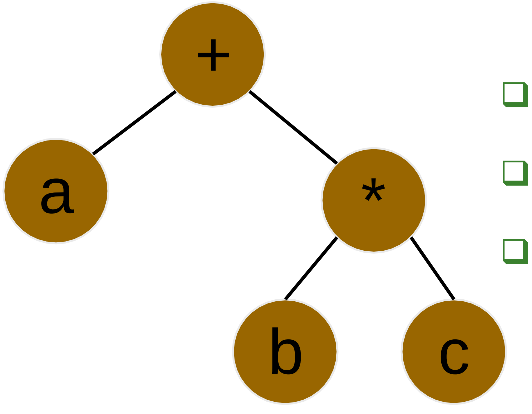
```
void pos_ordem(tree raiz){
    if (raiz != NULL){
        pos_ordem(raiz->esq);
        pos_ordem(raiz->dir);
        visita(raiz);
    }
}
```



Resultado: GDBHIEFCA

AB – Percursos

- Percurso para expressões aritméticas



  - Pré-ordem: +a*bc
  - In-ordem: a+(b*c)
  - Pós-ordem: abc*+

- Em algoritmos iterativos utiliza-se uma pilha ou um campo a mais em cada nó para guardar o nó anterior (pai)

Exercício 1 - em sala

- Dada uma ABB inicialmente vazia, insira (E DESENHE) os seguintes elementos (nessa ordem): 55, 64, 22, 48, 73, 17, 29, 36, 84, 54, 91, 44, 33, 12, 20, 83, 06. Após desenhar, responda as questões abaixo.
- A) Qual nó tem o maior grau e qual o grau árvore.
- B) Qual a altura/profundidade da árvore
- C) Usando a fórmula, qual a altura da árvore.
- D) É uma árvore balanceada? Sim/não e porque?
- E) Qual a sequência de nós no percurso Pré-ordem, Em ordem e Pós-ordem.

Exercício 2 - em sala

- Dada uma ABB inicialmente vazia, insira (E DESENHE) os seguintes elementos (nessa ordem): M, J, L, P, F, K, I, V, X, A, C, Q, R, B, S, U, T, D, Z, E, G, H, N, O, V. Após desenhar, responda as questões abaixo.
- A) Qual nó tem o maior grau e qual o grau árvore.
- B) Qual a altura/profundidade da árvore
- C) Usando a fórmula, qual a altura da árvore.
- D) Usamos a fórmula para saber a quantidade de nós da árvore? Sim/Não e porque?
- D) É uma árvore balanceada? Sim/não e porque?
- E) Qual a sequência de nós no percurso Pré-ordem, Em ordem e Pós-ordem.
- F) Represente a árvore com parênteses.
- G) Represente a árvore com os Diagramas de Venn

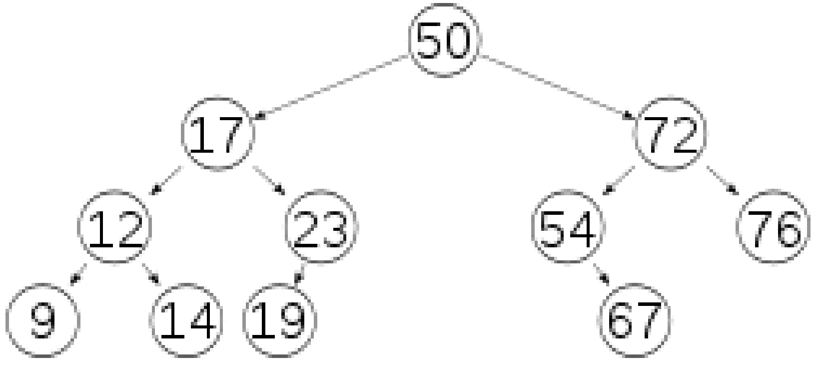
Arvore AVL

- A AVL (Adelson-Velskii e Landis – 1962) é uma árvore altamente balanceada, isto é, nas inserções e exclusões, procura-se executar uma rotina de balanceamento tal que as alturas das sub-árvores esquerda e sub-árvores direita tenham alturas **bem próximas**
- Definição
  - Uma árvore AVL é uma árvore na qual as alturas das sub-árvores esquerda e direita de cada nó diferem no máximo por uma unidade.
  - Fator de balanceamento
  - Altura da sub-árvore direita - altura da sub-árvore esquerda

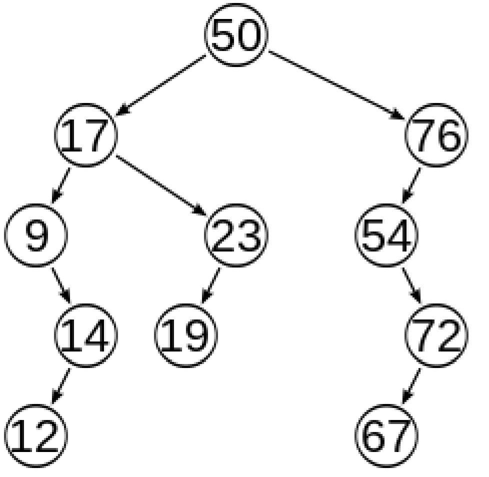
Arvore AVL

Complexidade de tempo em notação de big O		
Algoritmo	**Caso médio**	**Pior caso**
Espaço	O(n)	O(log n)
Busca	O(log n)	O(log n)
Inserção	O(log n)	O(log n)
Remoção	O(log n)	O(log n)

Arvore AVL



Árvore AVL



Árvore não AVL