Estácio

# ALGORITMOS E COMPLEXIDADE (ARA0174)

## AULA 12

DATA: 19/05/2022

# Aula 12- ALGORITMOS E COMPLEXIDADE

**TEMA 4** - Estruturas de dados dos tipos Árvore Binária e Árvore AVL


Estácio

# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## ÁRVORES – Conceitos Básicos

- Uma árvore consiste em um conjunto de elementos chamados de **NÓS ou VÉRTICES**. Em cada nó da árvore pode ser guardado qualquer tipo de informação.
- Um dos nós da árvore é chamado de **RAIZ**.
- Os demais nós formam m >= 0 conjuntos, onde cada um deles é uma árvore.

# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## ÁRVORES – Conceitos Básicos

- Cada nó da árvore é a raiz de uma subárvore. O número de subárvores de um nó é o **grau** daquele nó.

- Um nó de grau igual a zero é chamado **folha** ou **nó terminal**.

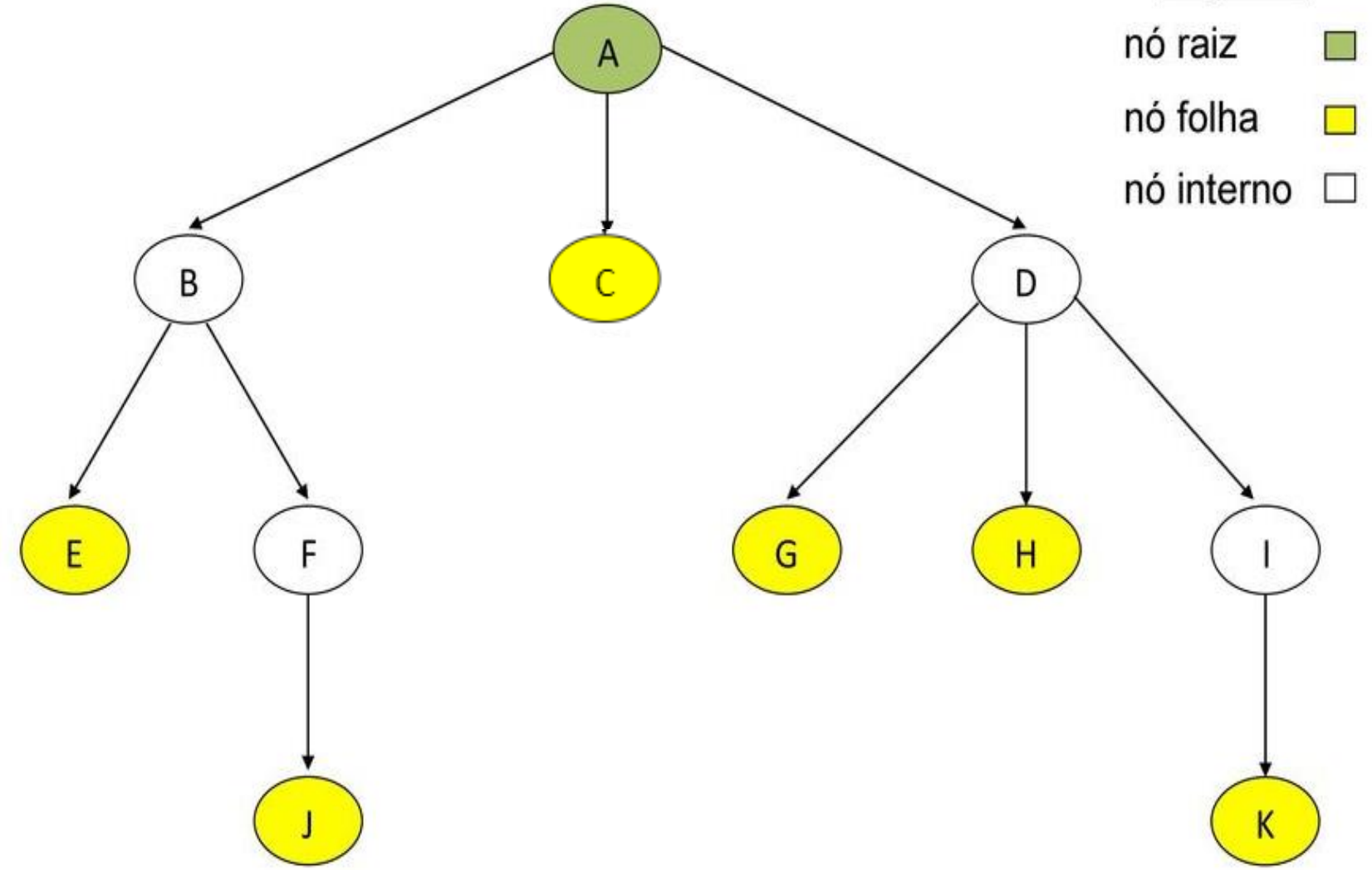
- O **nível** de um nó pode ser determinado da seguinte forma, a raiz é nível 0 e os demais tem um nível que é uma unidade maior que o nível de seu pai. Assim sendo nível é o número de nós do caminho mais curto que vai desde a raiz até esse nó.

TEMA 4

# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## ÁRVORES – Conceitos Básicos

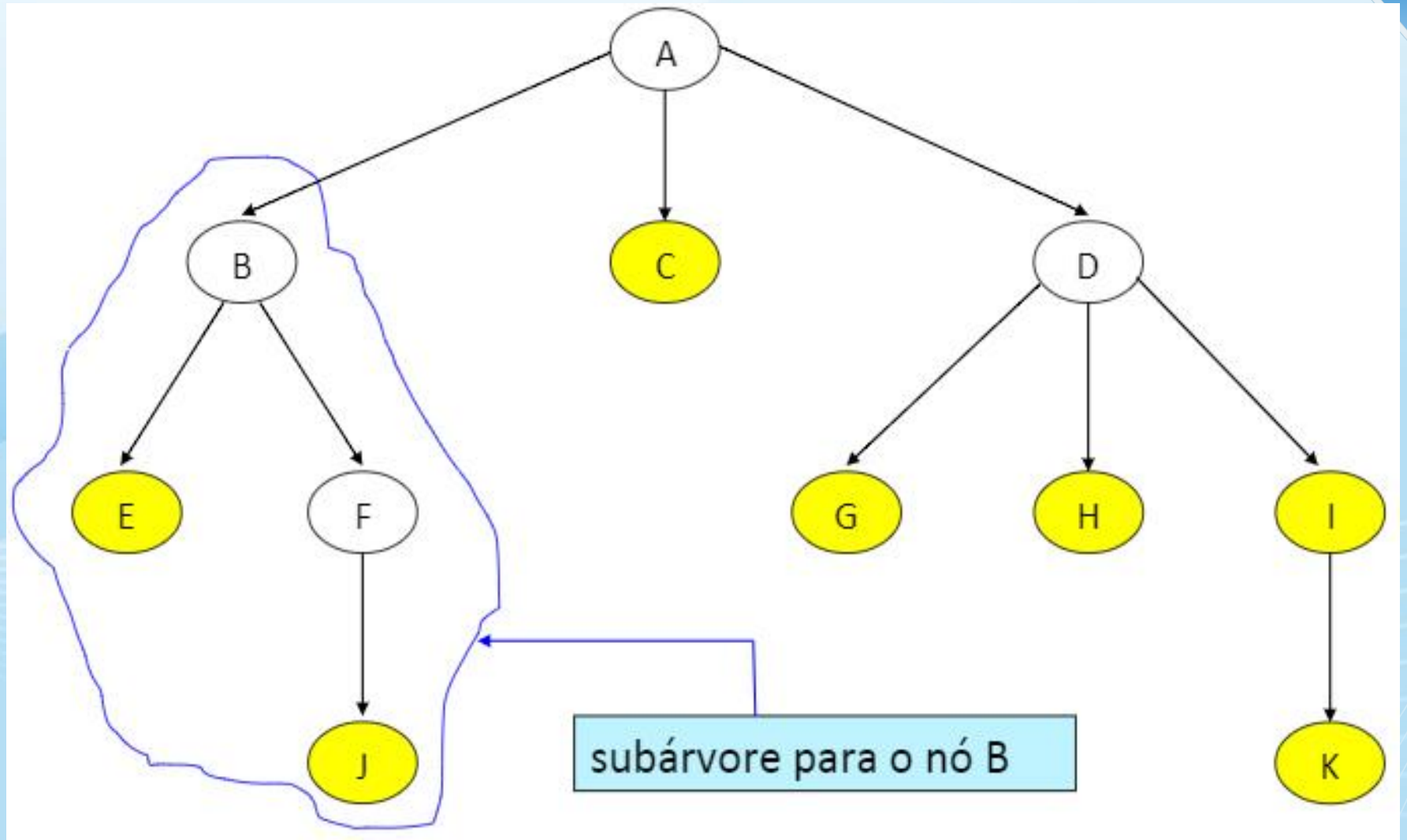


TEMA 4

# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## ÁRVORES – Conceitos Básicos

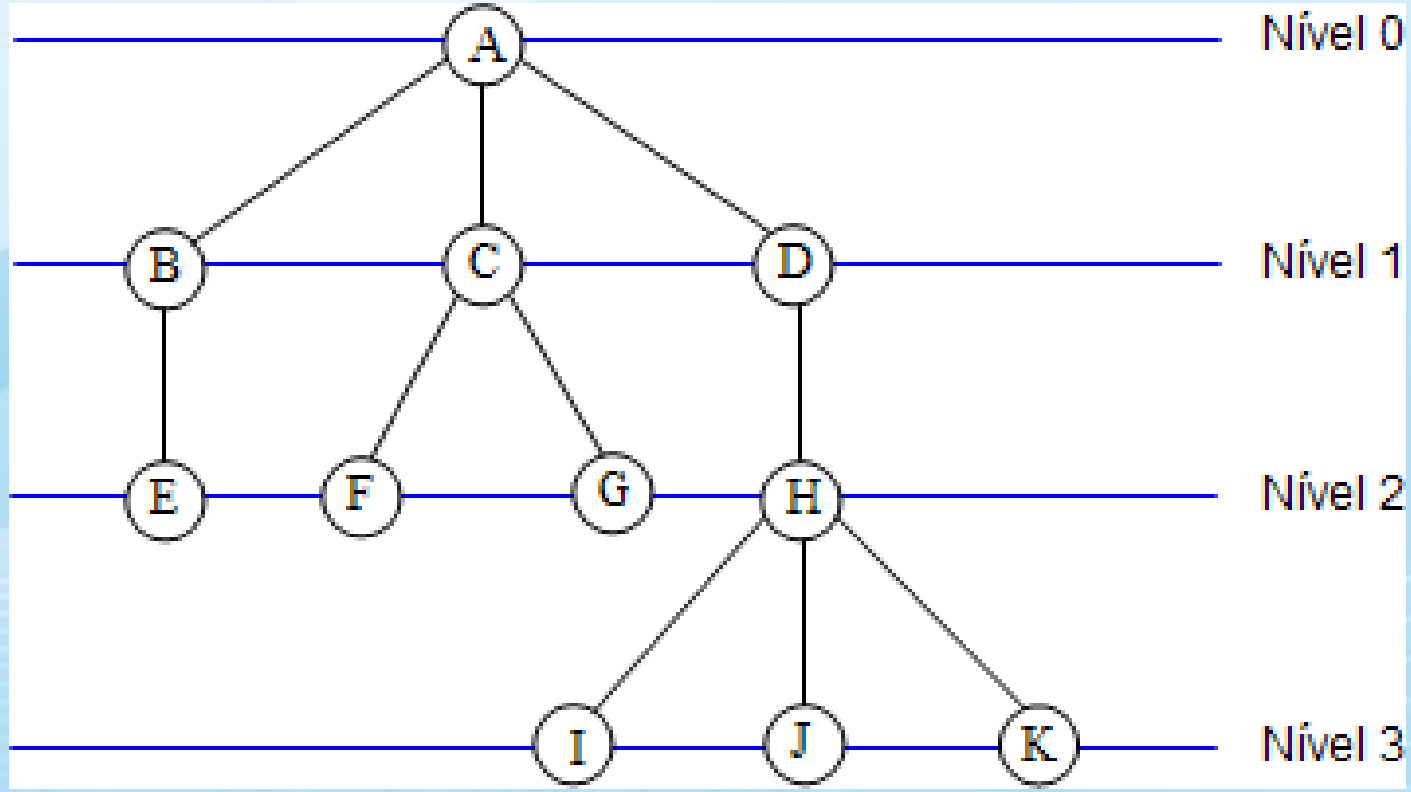


TEMA 4

# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## ÁRVORES – Conceitos Básicos



TEMA 4

# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## ÁRVORES – Representações

- Existem diversas maneiras de representar árvores. Uma representação que reflete a ideia de árvores como conjuntos aninhados é mostrado na figura abaixo.

**Grafo (representação mais utilizada)**

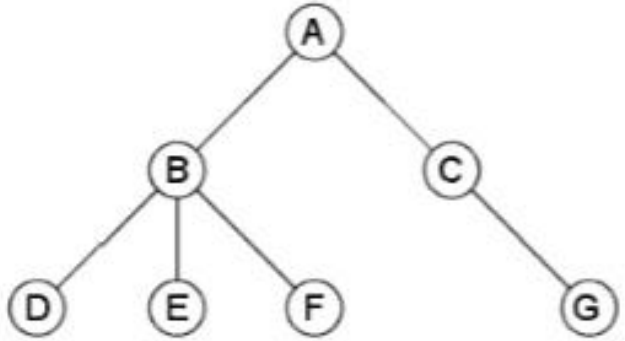
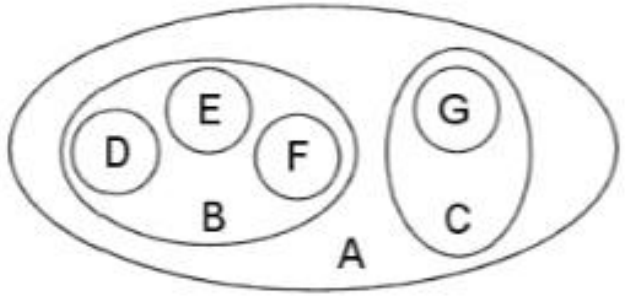



**Diagrama de Venn** (ou **digrama de inclusão** ou **conjuntos aninhados**)



TEMA 4

# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## ÁRVORES – Representações

**Grafo (representação mais utilizada)**

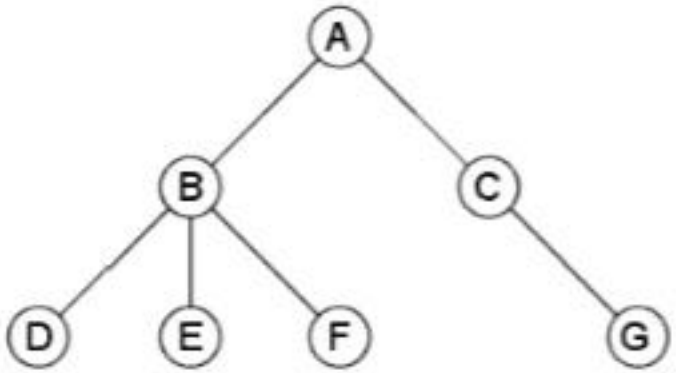
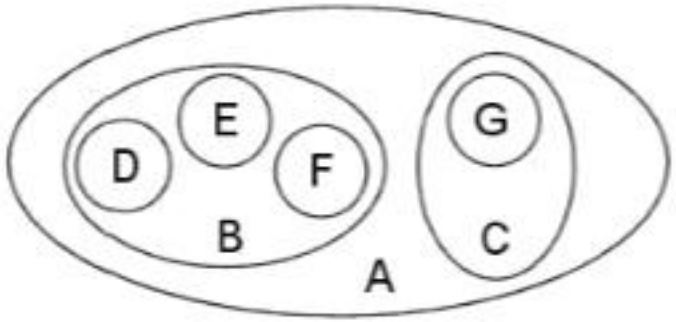



**Diagrama de Venn** (ou **digrama de inclusão** ou **conjuntos aninhados**)




**Identação**

A
B
D
E
F
C
G

**Parênteses Aninhados**

(A (B(D, E, F) C(G)))

TEMA 4

# Aula 12- ALGORITMOS E COMPLEXIDADE

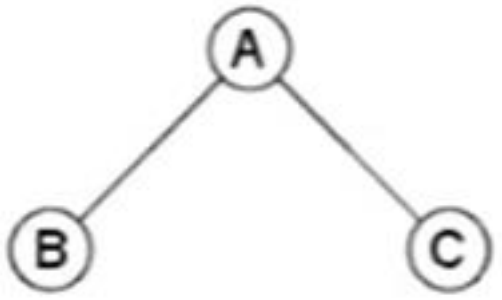
Estruturas de dados dos tipos Árvore Binária e Árvore AVL

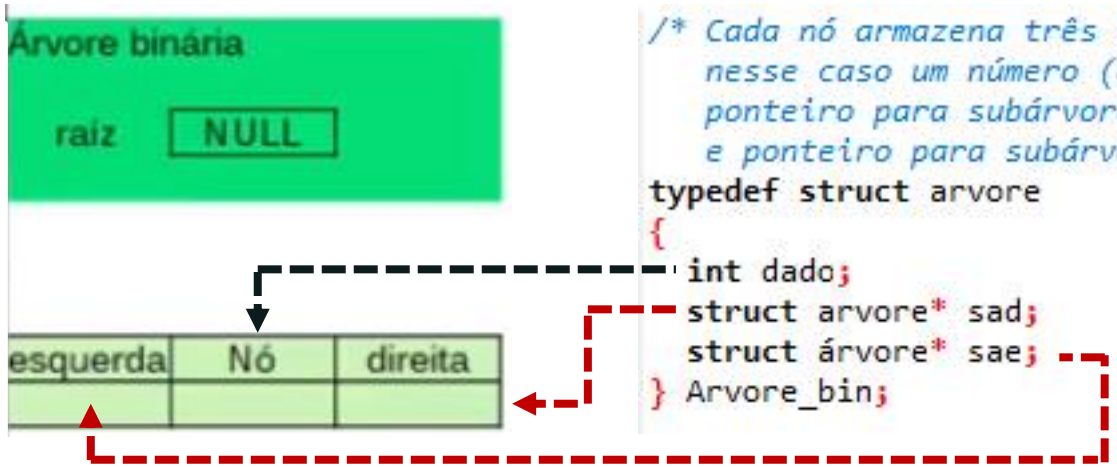
## ÁRVORES BINÁRIAS

Dentre as estruturas de dados dinâmicas, as árvores binárias estão entre as mais conhecidas, sendo uma estrutura muito eficiente para organização e busca de dados.

Sua principal característica, que justifica o nome "binária", advêm do fato de que cada nó pode possuir no máximo dois filhos, um à esquerda e um à direita.

**Representação (Definição básica de Árvore)**




```
/* Cada nó armazena três informações:
   nesse caso um número (dado),
   ponteiro para subárvore à direita (sad)
   e ponteiro para subárvore à esquerda (sae).*/
typedef struct arvore
{
   int dado;
   struct arvore* sad;
   struct árvore* sae;
} Arvore_bin;
```

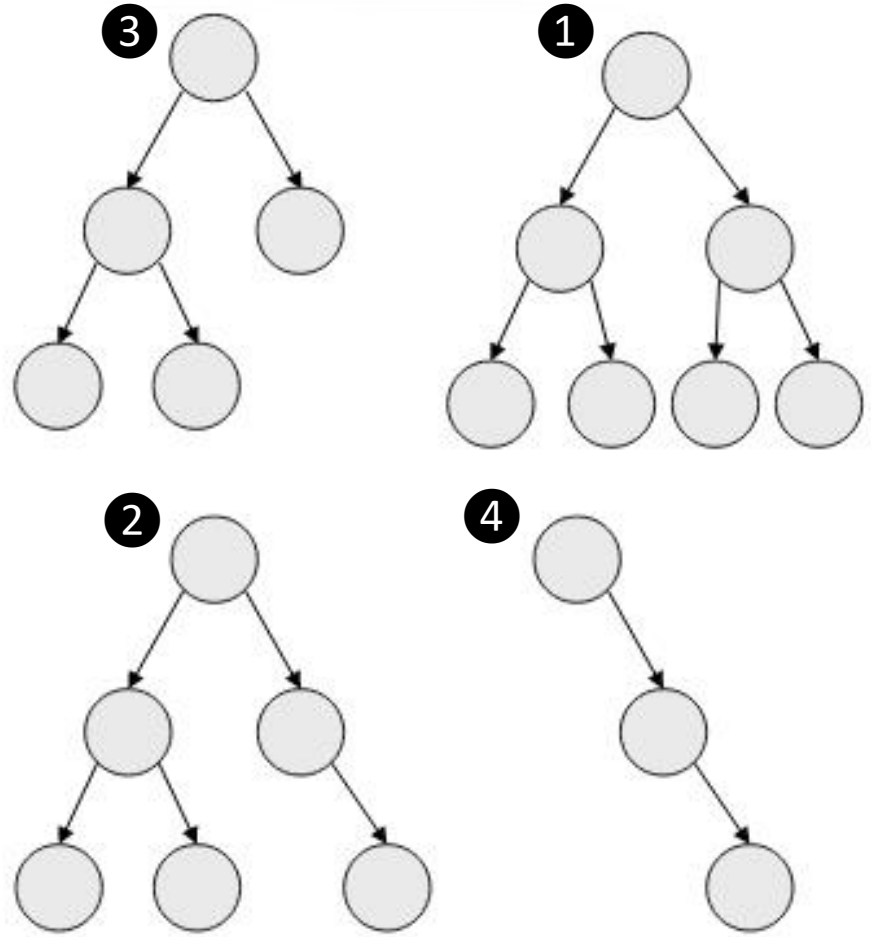
TEMA 4

# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## ÁRVORES – Representações

1. **Estritamente Binária** : todo nó interno tem exatamente 2 filhos.
2. **Completa (Cheia)** : estritamente binária, e todas as folhas estão no mesmo nível.
3. **Balanceada** : para todo nó, a diferença de altura de suas sub-árvores é de no máximo 1.
4. **Degenerada** : todos os nós têm no máximo 1 filho (lista encadeada).



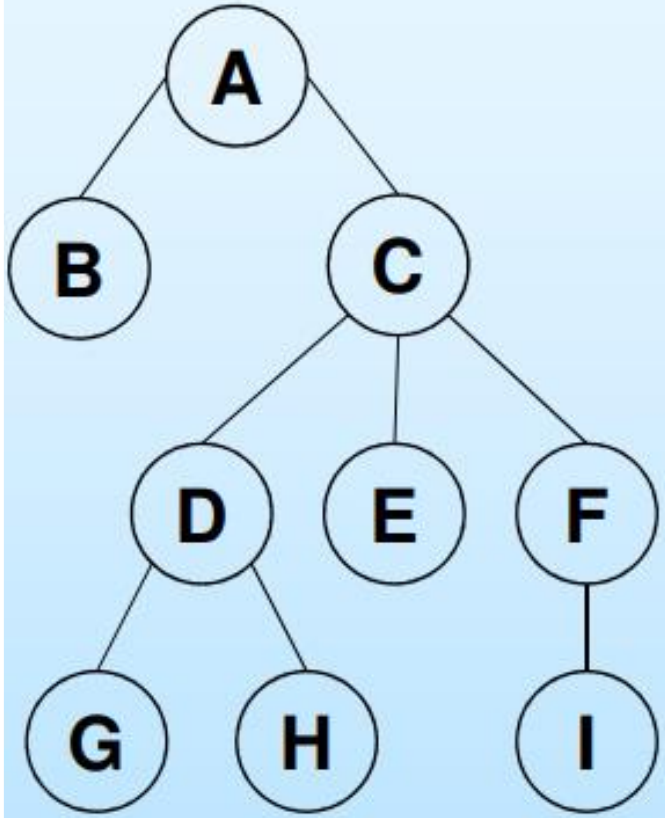
TEMA 4

# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## EXERCÍCIO 1

**Informe o grau de cada nó, quem são as folhas e quem são os nós internos.**




- **Graus dos nós**
  - G(A)=
  - G(B)=
  - G(C)=
  - G(D)=
  - G(E)=
  - G(F)=
  - G(G)=
  - G(H)=
  - G(I)=

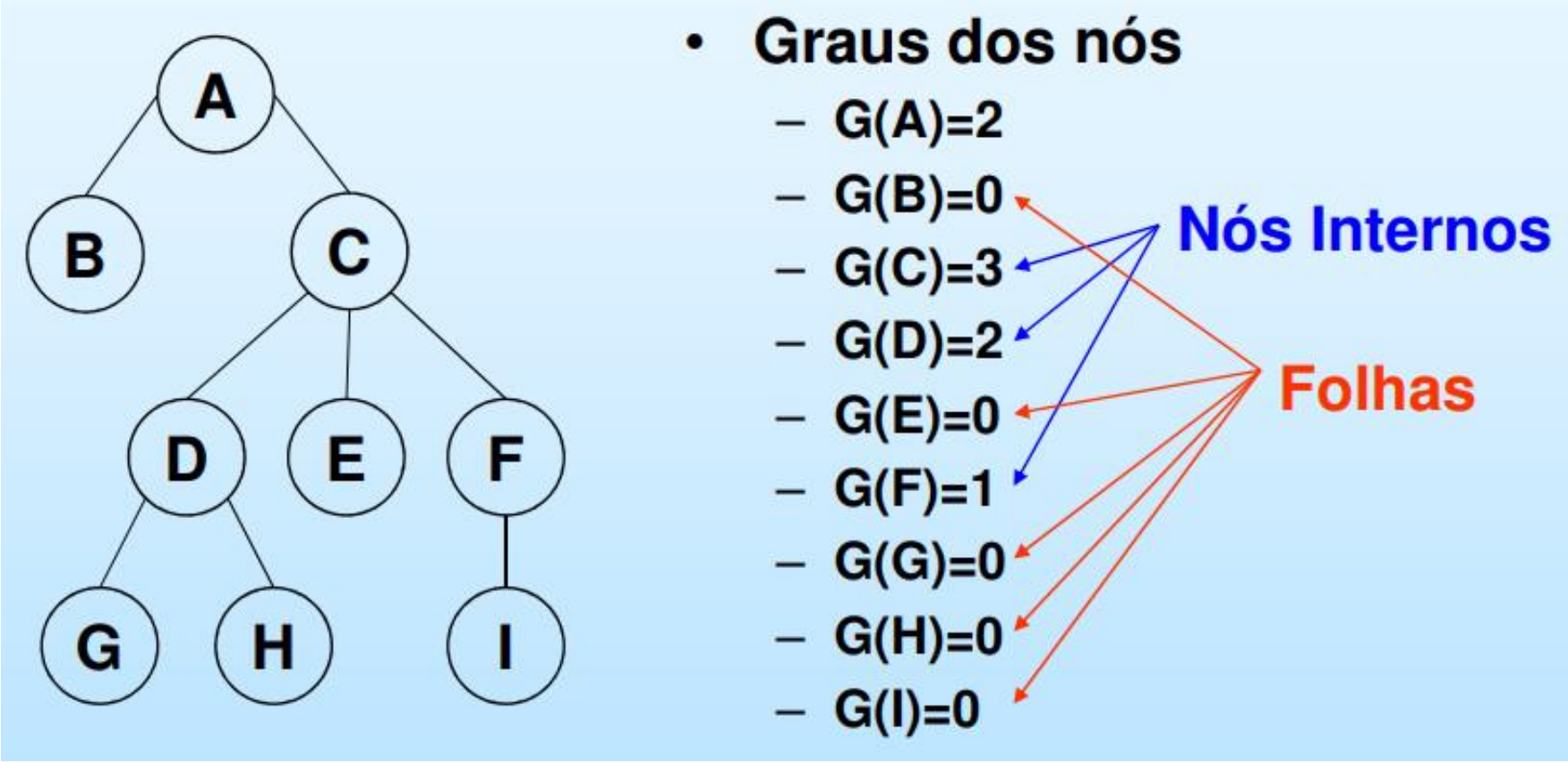
TEMA 4

# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## EXERCÍCIO 1

**Informe o grau de cada nó, quem são as folhas e quem são os nós internos.**



TEMA 4

# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## EXERCÍCIO 2

Sobre estruturas de dados do tipo árvore binária, analise as assertivas abaixo.

I. Diferente das listas simplesmente encadeadas, as árvores binárias permitem que cada nó tenha dois nós sucessores (filhos).

II. Raiz (*root*) é o nó mais inferior da árvore binária que não possui sucessores (filhos).

III. Folha (*leaf*) é qualquer nó da árvore binária que não tenha sucessores (filhos).

É correto o que se afirma em:

(  ) I, II e III.

(  ) I, apenas.

(  ) II, apenas.

(  ) II e III, apenas.

(  ) I e III, apenas.

TEMA 4

# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## EXERCÍCIO 2

Sobre estruturas de dados do tipo árvore binária, analise as assertivas abaixo.

I. Diferente das listas simplesmente encadeadas, as árvores binárias permitem que cada nó tenha dois nós sucessores (filhos).

II. Raiz (*root*) é o nó mais inferior da árvore binária que não possui sucessores (filhos).

III. Folha (*leaf*) é qualquer nó da árvore binária que não tenha sucessores (filhos).

É correto o que se afirma em:

(  ) I, II e III.

(  ) I, apenas.

(  ) II, apenas.

(  ) II e III, apenas.

( **X** ) I e III, apenas.

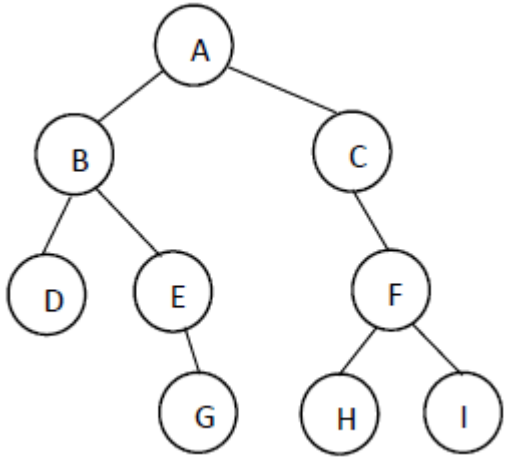
TEMA 4

# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## EXERCÍCIO 3

**Analise a seguinte árvore binária e assinale a alternativa correta.**




( ) “A” é filho de todos.
( ) “B” e “C” são caules da árvore.
( ) “B” tem grau de saída 3 e “C” grau 2.
( ) Árvore enraizada em “A”.
( ) Com exceção do nó “A”, que é raiz, os demais nós são conhecido como folhas.

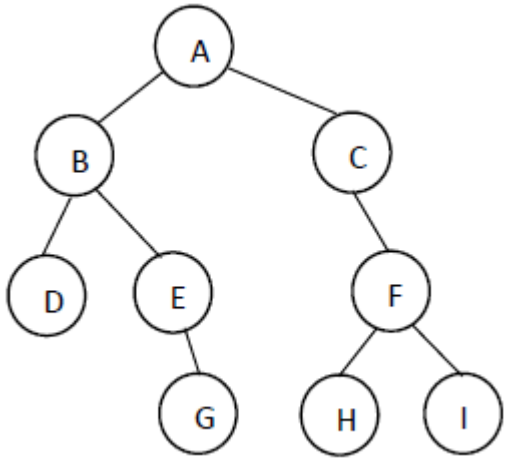
TEMA 4

# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## EXERCÍCIO 3

**Analise a seguinte árvore binária e assinale a alternativa correta.**




(  ) “A” é filho de todos.
(  ) “B” e “C” são caules da árvore.
(  ) “B” tem grau de saída 3 e “C” grau 2.
( X ) Árvore enraizada em “A”.
(  ) Com exceção do nó “A”, que é raiz, os demais nós são conhecido como folhas.

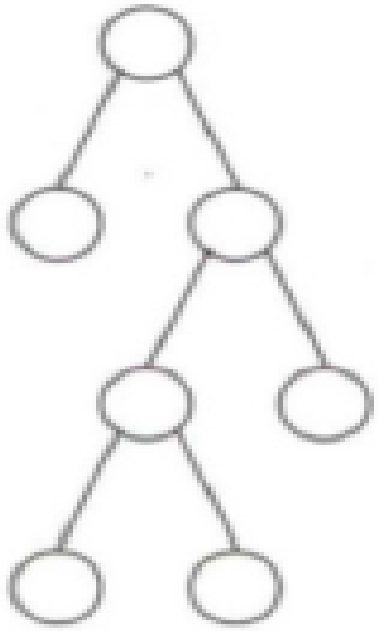
TEMA 4

# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## EXERCÍCIO 4

**A imagem a seguir representa uma estrutura de dados chamada árvore binária. Há vários tipos de árvores binárias. Qual é o tipo de árvore binária que tal imagem representa?**



( ) Árvore binária em largura.
( ) Árvore binária em profundidade.
( ) Árvore binária cheia.
( ) Árvore binária completa.
( ) Árvore estritamente binária.

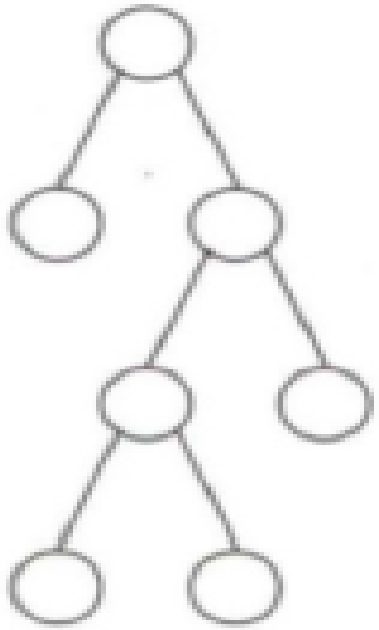
TEMA 4

# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## EXERCÍCIO 4

**A imagem a seguir representa uma estrutura de dados chamada árvore binária. Há vários tipos de árvores binárias. Qual é o tipo de árvore binária que tal imagem representa?**



( ) Árvore binária em largura.
( ) Árvore binária em profundidade.
( ) Árvore binária cheia.
( ) Árvore binária completa.
( **X** ) Árvore estritamente binária.

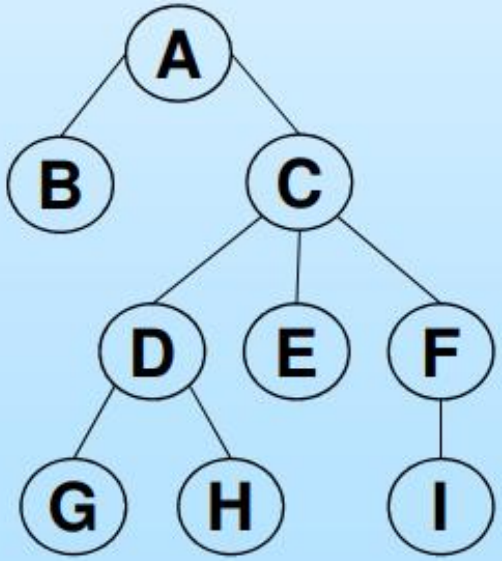
TEMA 4

# Aula 12- ALGORITMOS E COMPLEXIDADE

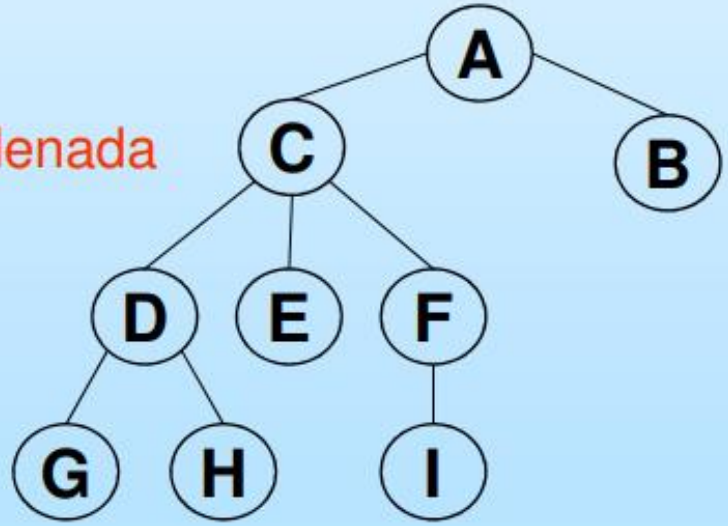
Estruturas de dados dos tipos Árvore Binária e Árvore AVL

- Árvore Ordenada
  - Os filhos de cada nó estão ordenados (assume-se ordenação da esquerda para a direita)

ordenada




desordenada



TEMA 4

# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## Árvores Binárias

Árvore Binária: Uma árvore binária é uma árvore que pode ser nula, ou então tem as seguintes características:

- Existe um nó especial denominado raiz;
- Nenhum nó tem grau superior a 2 (dois), isto é, nenhum nó tem mais de dois filhos;
- Existe um “senso de posição”, ou seja, distingue-se entre uma subárvore esquerda e uma subárvore direita.

TEMA 4

# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## Árvores Binárias

Atravessamento (ou caminhamento) de árvore é a passagem de forma sistemática por cada um de seus nós;

Diferentes formas de percorrer os nós de uma árvore:

- Pré-ordem ou prefixa (busca em profundidade);
- Em ordem ou infixa (ordem central);
- Pós-ordem ou posfixa;
- Em nível.

TEMA 4

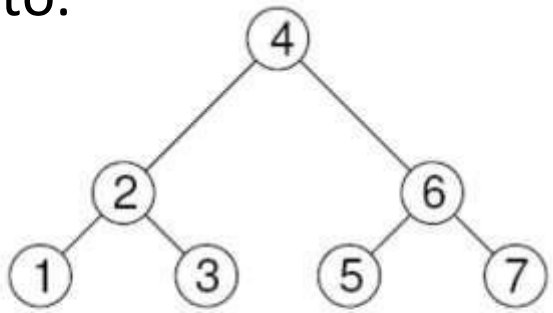
# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## Árvores Binárias

**Pré-ordem (prefixa)**

- visitar a raiz;
- caminhar na subárvore à esquerda, segundo este caminhamento;
- caminhar na subárvore à direita, segundo este caminhamento.




4 2 1 3 6 5 7

TEMA 4

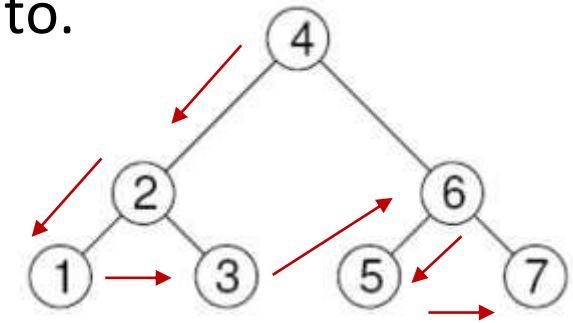
# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## Árvores Binárias

**Pré-ordem (prefixa)**

- visitar a raiz;
- caminhar na subárvore à esquerda, segundo este caminhamento;
- caminhar na subárvore à direita, segundo este caminhamento.




4 2 1 3 6 5 7

TEMA 4

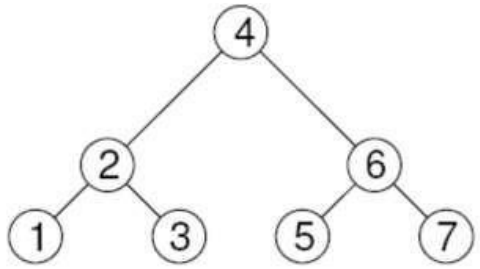
# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## Árvores Binárias

**Atravessamento em ordem (infixa)**

- caminhar na subárvore à esquerda, segundo este caminhamento;
- visitar a raiz;
- caminhar na subárvore à direita, segundo este caminhamento




1 2 3 4 5 6 7

TEMA 4

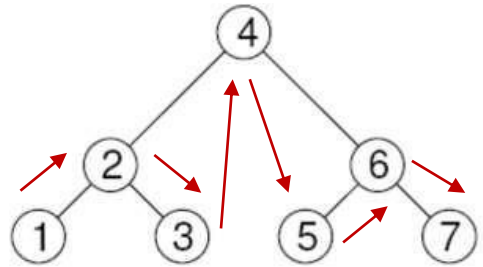
# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## Árvores Binárias

**Atravessamento em ordem (infixa)**

- caminhar na subárvore à esquerda, segundo este caminhamento;
- visitar a raiz;
- caminhar na subárvore à direita, segundo este caminhamento




1 2 3 4 5 6 7

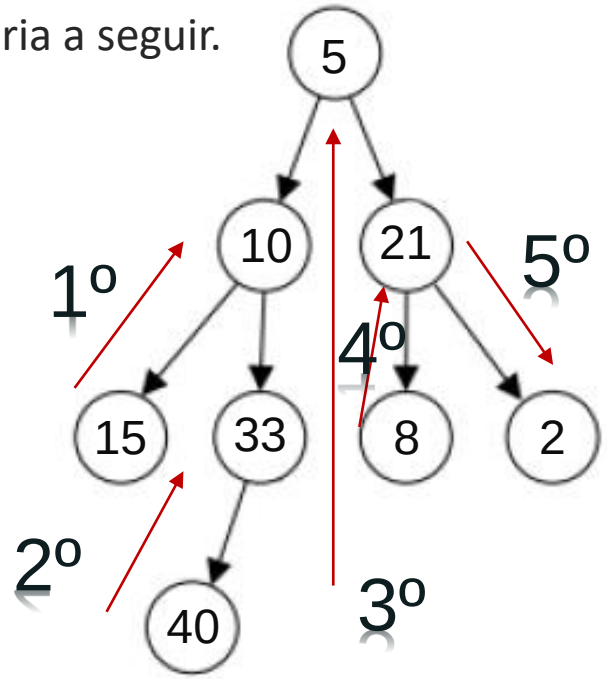
TEMA 4

# Aula 12- ALGORITMOS E COMPLEXIDADE

## Estruturas de dados dos tipos Árvore Binária e Árvore AVL

### OUTRO EXEMPLO

Observe a Árvore Binária a seguir.




Escreva a sequência de visitação sobre essa árvore pelo caminhamento central (infixado).

**15 – 10 – 40 – 33 – 5 – 8 – 21 – 2.**

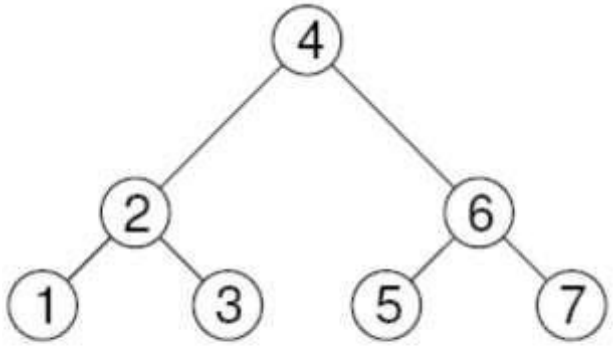
# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## Árvores Binárias

**Atravessamento pós-ordem (posfixa)**

- a) caminhar na subárvore à esquerda, segundo este caminhamento;
- b) caminhar na subárvore à direita, segundo este caminhamento;
- c) visitar a raiz.




1 3 2 5 7 6 4

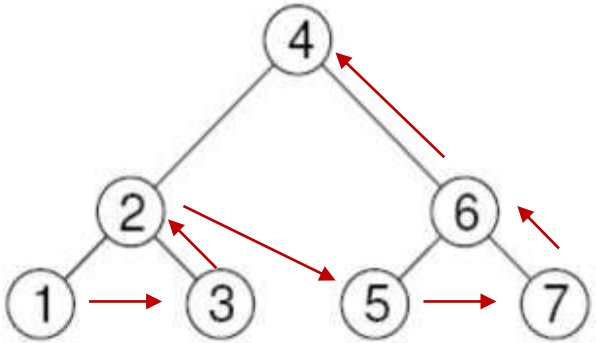
# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## Árvores Binárias

**Atravessamento pós-ordem (posfixa)**

- a) caminhar na subárvore à esquerda, segundo este caminhamento;
- b) caminhar na subárvore à direita, segundo este caminhamento
- c) visitar a raiz.




1 3 2 5 7 6 4

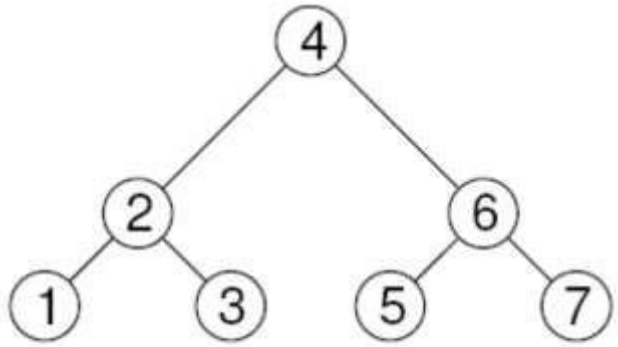
# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## Árvores Binárias

**Atravessamento em nível**

- Percorre-se a árvore de cima para baixo e da direita para a esquerda.




4 2 6 1 3 5 7

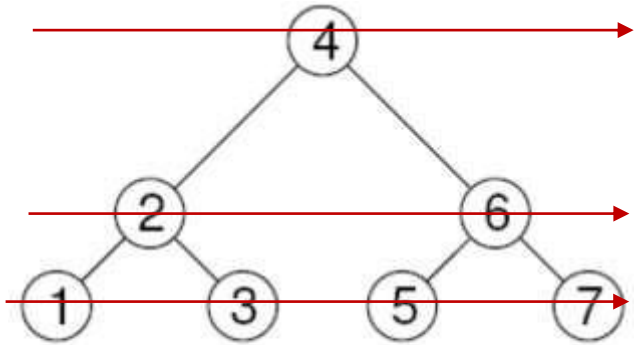
# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## Árvores Binárias

**Atravessamento em nível**

- Percorre-se a árvore de cima para baixo e da direita para a esquerda.




4 2 6 1 3 5 7

TEMA 4

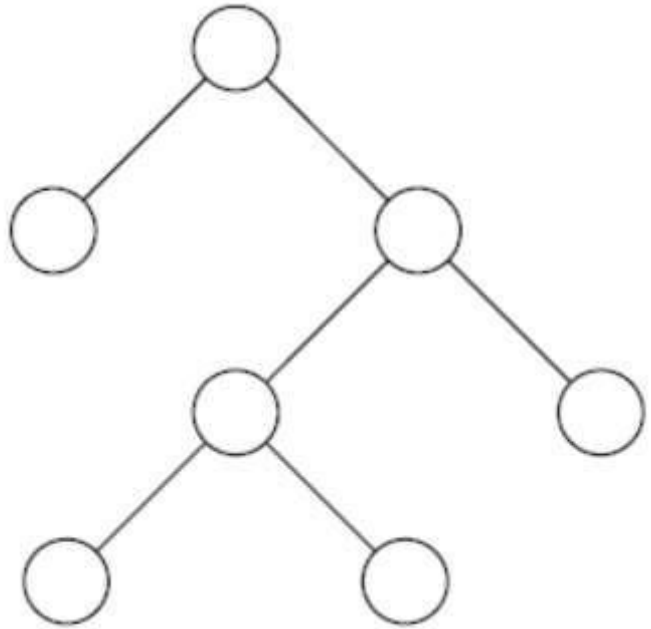
# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## Árvores Binárias

**Árvore Estritamente Binária:**

- É uma árvore binária na qual todo nó tem 0 ou 2 subárvores, ou seja, nenhum nó tem “filho  único”.



TEMA 4

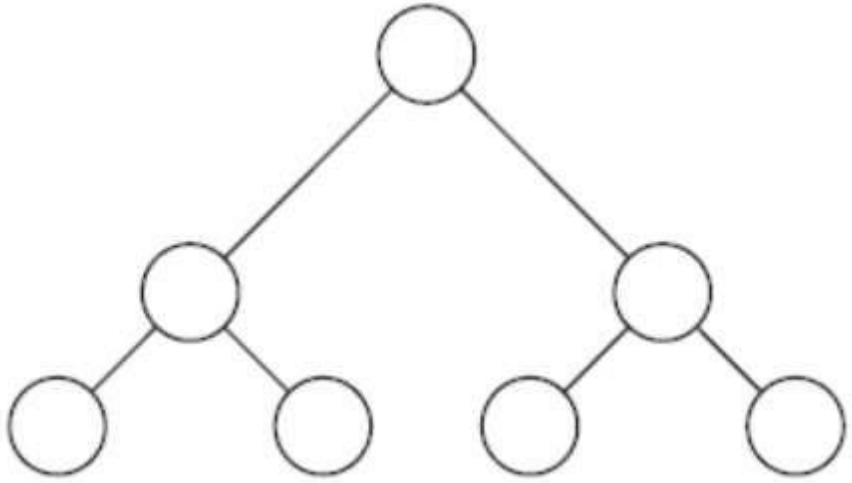
# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## Árvores Binárias

**Árvore Binária Cheia**

- Todos os nós, exceto os do último nível, possuem exatamente duas subárvores.
- Uma árvore binária cheia de altura h tem 2h – 1 nós.



TEMA 4

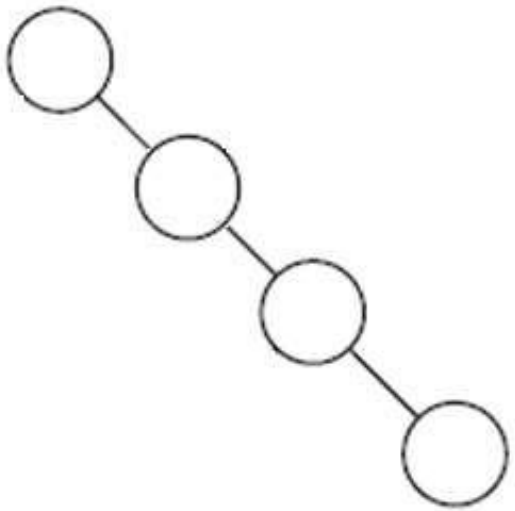
# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## Árvores Binárias

**Árvore Degenerada**

- Cada nó possui exatamente um filho, e a árvore tem o mesmo número de níveis que de nós



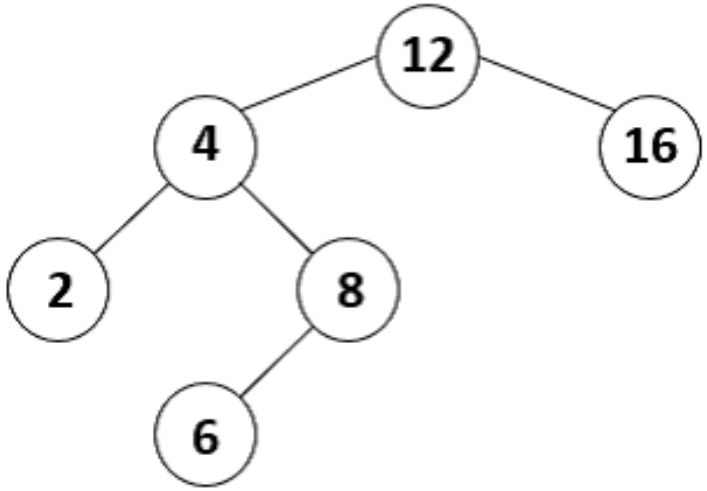
TEMA 4

# Aula 12- ALGORITMOS E COMPLEXIDADE

## Estruturas de dados dos tipos Árvore Binária e Árvore AVL

### EXERCÍCIO 5

Considere a árvore binária da figura a seguir:




Os resultados das consultas dos nós dessa árvore binária em pré-ordem e pós-ordem são, respectivamente:

(  ) (2 4 6 8 12 16) e (2 6 8 4 16 12).
(  ) (12 4 2 8 6 16) e (2 4 6 8 12 16).
(  ) (2 6 8 4 16 12) e (12 4 2 8 6 16).
(  ) (2 4 6 8 12 16) e (12 4 2 8 6 16).
(  ) (12 4 2 8 6 16) e (2 6 8 4 16 12).

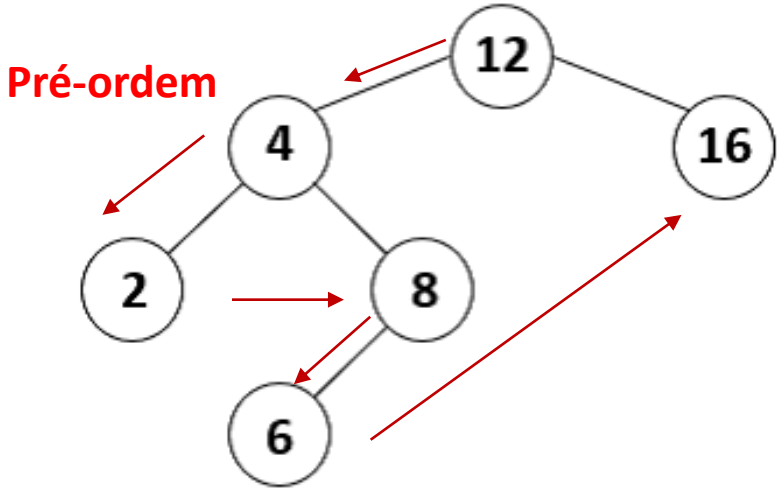
TEMA 4

# Aula 12- ALGORITMOS E COMPLEXIDADE

## Estruturas de dados dos tipos Árvore Binária e Árvore AVL

### EXERCÍCIO 5

Considere a árvore binária da figura a seguir:




Os resultados das consultas dos nós dessa árvore binária em pré-ordem e pós-ordem são, respectivamente:

(  ) (2 4 6 8 12 16) e (2 6 8 4 16 12).
(  ) (12 4 2 8 6 16) e (2 4 6 8 12 16).
(  ) (2 6 8 4 16 12) e (12 4 2 8 6 16).
(  ) (2 4 6 8 12 16) e (12 4 2 8 6 16).
( **X** ) (12 4 2 8 6 16) e (2 6 8 4 16 12).

Em **pré-ordem** percorre-se primeiro visitando a raiz e depois a sub-árvore da esquerda depois da direita. Ou seja: 12, 4, 2, 8 ,6 16
Em **pós-ordem** a raiz é a ultima a ser visitada: Sendo assim: 2,6,8,4,16,12

TEMA 4

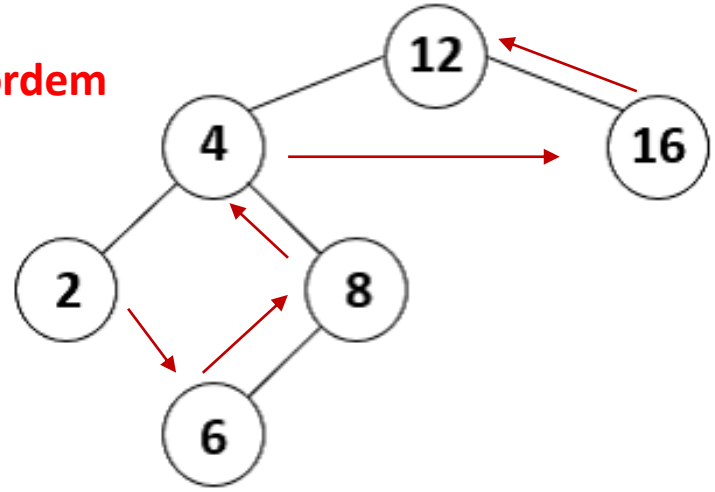
# Aula 12- ALGORITMOS E COMPLEXIDADE

## Estruturas de dados dos tipos Árvore Binária e Árvore AVL

### EXERCÍCIO 5

Considere a árvore binária da figura a seguir:

Pós-ordem




Os resultados das consultas dos nós dessa árvore binária em pré-ordem e pós-ordem são, respectivamente:

(  ) (2 4 6 8 12 16) e (2 6 8 4 16 12).
(  ) (12 4 2 8 6 16) e (2 4 6 8 12 16).
(  ) (2 6 8 4 16 12) e (12 4 2 8 6 16).
(  ) (2 4 6 8 12 16) e (12 4 2 8 6 16).
( **X** ) (12 4 2 8 6 16) e (2 6 8 4 16 12).

Em **pré-ordem** percorre-se primeiro visitando a raiz e depois a sub-árvore da esquerda depois da direita. Ou seja: 12, 4, 2, 8 ,6 16
Em **pós-ordem** a raiz é a ultima a ser visitada: Sendo assim: 2,6,8,4,16,12

TEMA 4

# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## EXERCÍCIO 6

A operação de destruição de uma árvore requer um tipo de percurso em que a liberação de um nó é realizada apenas após todos os seus descendentes terem sido também liberados. Segundo essa descrição, a operação de destruição de uma árvore deve ser implementada utilizando o percurso.

(  ) em ordem.
(  ) pré-ordem.
(  ) central.
(  ) simétrico.
(  ) pós-ordem.

TEMA 4

# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## EXERCÍCIO 6

A operação de destruição de uma árvore requer um tipo de percurso em que a liberação de um nó é realizada apenas após todos os seus descendentes terem sido também liberados. Segundo essa descrição, a operação de destruição de uma árvore deve ser implementada utilizando o percurso.

(  ) em ordem.
(  ) pré-ordem.
(  ) central.
(  ) simétrico.
( **X** ) pós-ordem.

Considerando que a raiz das árvores contém o ponteiro para os dois nós associados (sub-árvore da esquerda e sub-árvore da direita) então é necessário visitar todas subárvores e liberando os nós folha e posteriormente os nós raiz. Ou seja, esse percurso deve deixar a raiz por último, caracterizando o percurso pós-ordem.

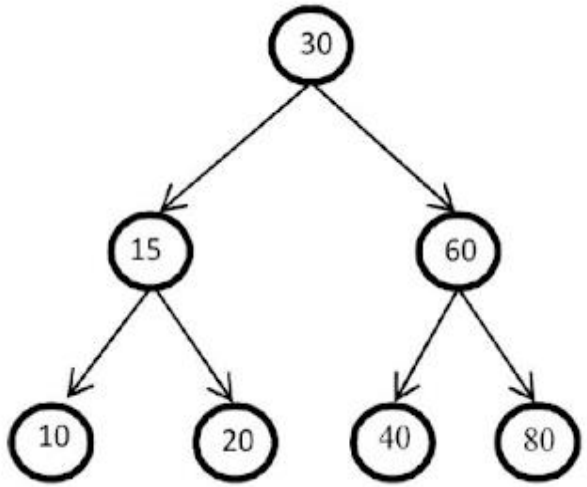
TEMA 4

# Aula 12- ALGORITMOS E COMPLEXIDADE

## Estruturas de dados dos tipos Árvore Binária e Árvore AVL

### EXERCÍCIO 7

Observe a Árvore Binária de Busca (ABB) a seguir.




Assinale a alternativa que apresenta, corretamente, a sequência de inserção que gera essa ABB.

(  ) 30, 15, 40, 10, 20, 60, 80
(  ) 30, 15, 40, 10, 20, 80, 60
(  ) 30, 15, 60, 10, 20, 40, 80
(  ) 30, 60, 20, 80, 15, 10, 40
(  ) 30, 60, 40, 10, 20, 15, 80

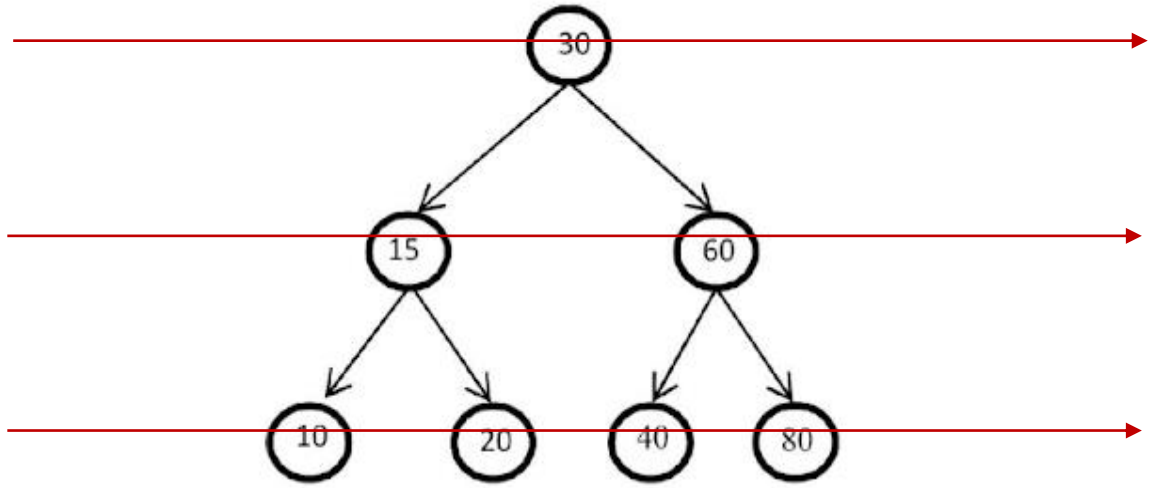
TEMA 4

# Aula 12- ALGORITMOS E COMPLEXIDADE

## Estruturas de dados dos tipos Árvore Binária e Árvore AVL

### EXERCÍCIO 7

Observe a Árvore Binária de Busca (ABB) a seguir.




Assinale a alternativa que apresenta, corretamente, a sequência de inserção que gera essa ABB.

(  ) 30, 15, 40, 10, 20, 60, 80
(  ) 30, 15, 40, 10, 20, 80, 60
( **X** ) 30, 15, 60, 10, 20, 40, 80
(  ) 30, 60, 20, 80, 15, 10, 40
(  ) 30, 60, 40, 10, 20, 15, 80

É preciso inserir os nós raiz primeiro, a seguir inserir as folhas.

# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## Árvores AVL

- Árvores Balanceadas
- Balanceamento estático e dinâmico!
- Árvores AVL
- Fator de Balanceamento (Fatbal)
- Rotação Simples(Esquerda e direita)
- Rotação Dupla (Esquerda e Direita)
- Exemplos
- Referências.

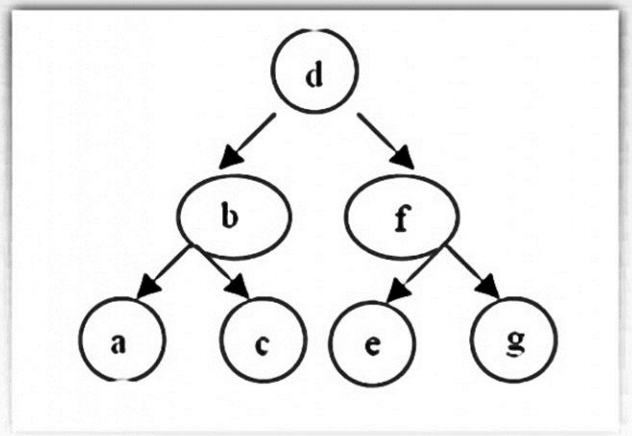
TEMA 4

# Aula 12- ALGORITMOS E COMPLEXIDADE

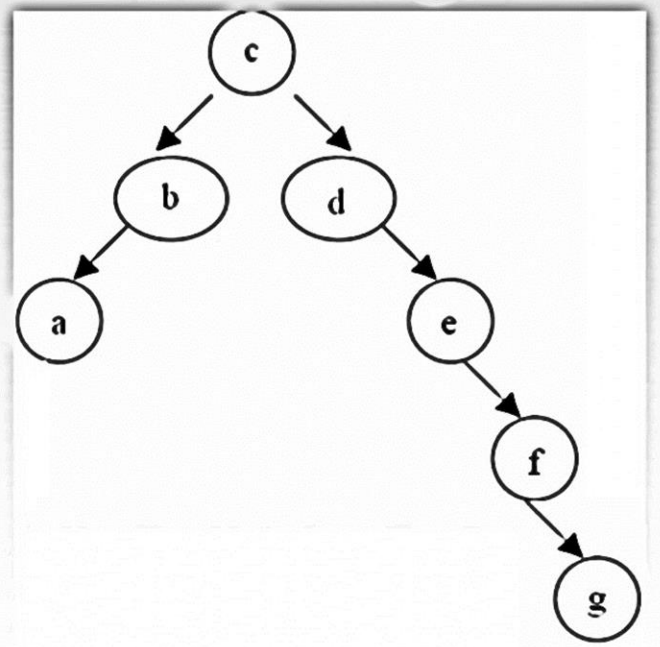
Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## Árvores
### Balanceadas

- Uma arvore é considerada balanceada quando suas sub-arvores à esquerda e à direita possuem a mesma altura.
- A árvore não balanceada é definida como degenerada




Árvore Binária Balanceada




Árvore Binária Degenerada

TEMA 4

# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## Árvores Balanceadas

- **Balanceamento Estático:**

  - Este balanceamento consiste em, depois de um certo tempo de uso da árvore, destruir sua estrutura, guardando suas informações em uma lista ordenada e reconstruí-la de forma balanceada.

- **Balanceamento Dinâmico:**

  - Tem por objetivo reajustar os nós de uma árvore sempre que uma inserção ou remoção provocar desbalanceamento.

  - Um exemplo de Balanceamento dinâmico são as árvores AVL.

TEMA 4

# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## Árvores AVL

- O termo AVL vem de seus fundadores Adel´son, Vel´skii e Landis (1962). Foi a primeira estrutura de dados a oferecer operações de inserção, remoção e busca em tempo logaritmo ou seja é um **algoritmo muito rápido**.

- Em uma árvore degenerada de 10.000 nós, são necessárias 5.000 comparações para efetuar uma busca, já numa árvore AVL, com o mesmo número de nós, essa média baixa para 14 comparações. —

- A árvore AVL é uma árvore binária de busca e sua estrutura foi construída de forma que a altura da sub-árvore direita é diferente da altura da sub-árvore esquerda de no máximo 1.

# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## Árvores AVL

## *Fator de Balanceamento*

- Sendo assim, para cada nó define-se um fator de balanceamento(fatbal), que deve ser -1,0 ou 1.

**Fatbal = altura (sub-arvore direita) – altura (sub-árvore esquerda)**

*-> Fatbal = -1, quando a sub-árvore da esquerda é um nível mais alto que a direita.*

*-> Fatbal = 0, quando as duas sub-árvores tem a mesma altura.*

*-> Fatbal = 1, quando a sub-árvore da direita é um nível mais alto que a esquerda.*

# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## Balanceamento em AVL

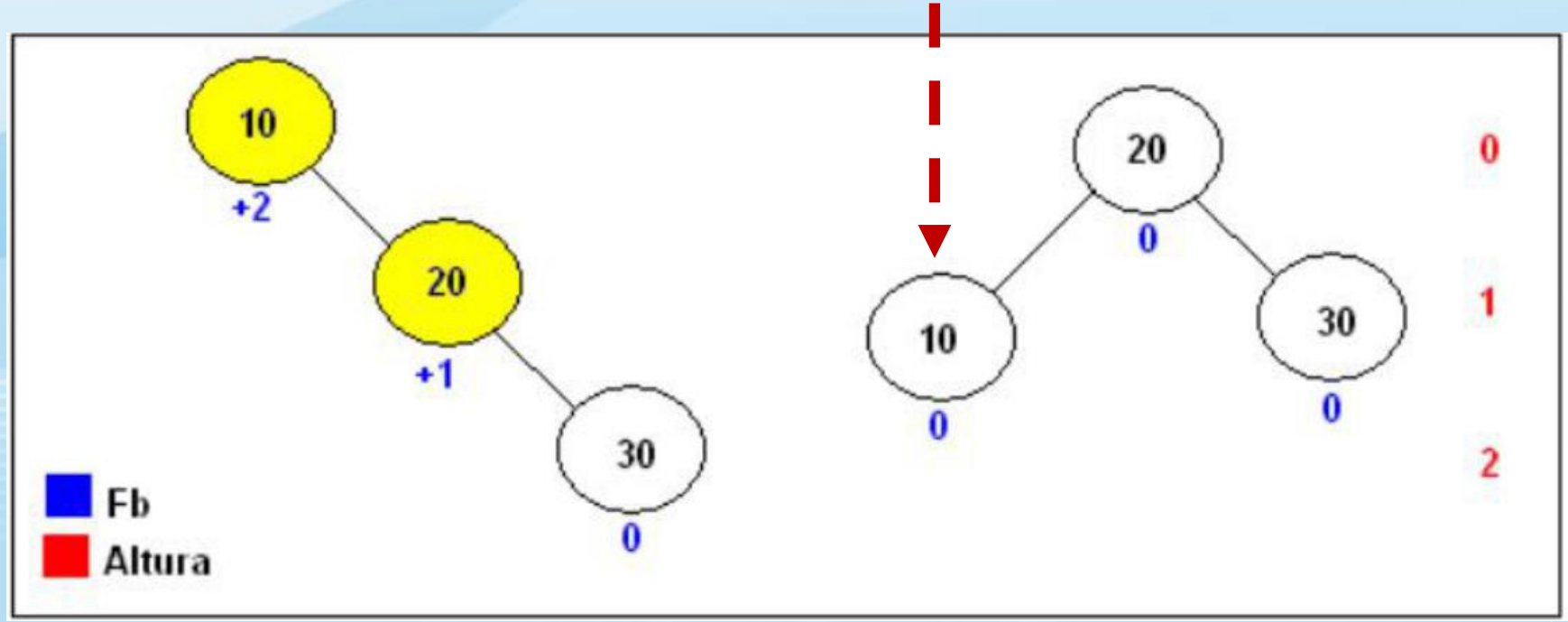
- Inserimos um novo nodo na árvore.
- Esta inserção pode ou não alterar as propriedades de balanceamento.
- Caso a inserção desse novo nodo não viole alguma propriedade de balanceamento, podemos continuar inserindo novos nodos.
- Se a inserção afetar as propriedades de balanceamento devemos restaurar o balanço da árvore. Esta restauração é efetuada através de ROTAÇÕES na árvore.

# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## ROTAÇÃO:

## Rotação simples a esquerda

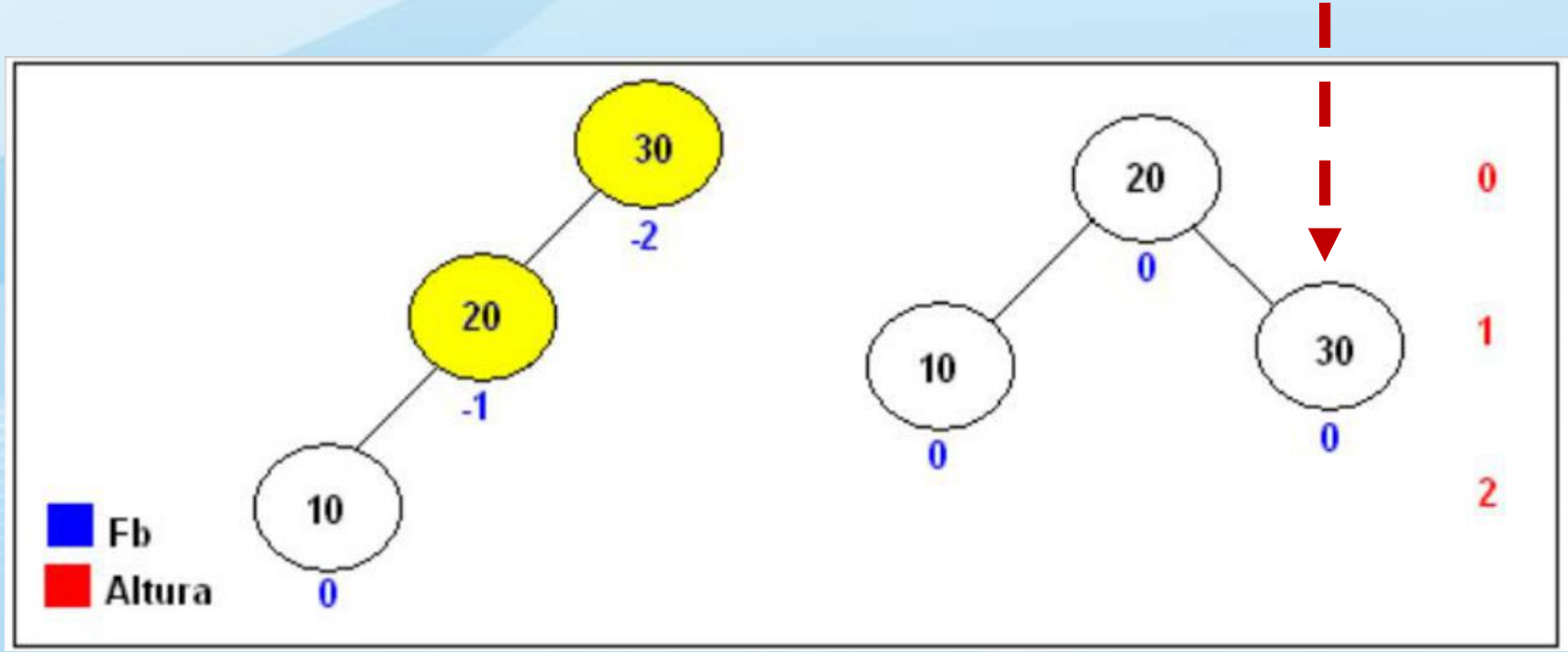


# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## ROTAÇÃO:

## Rotação simples a direita

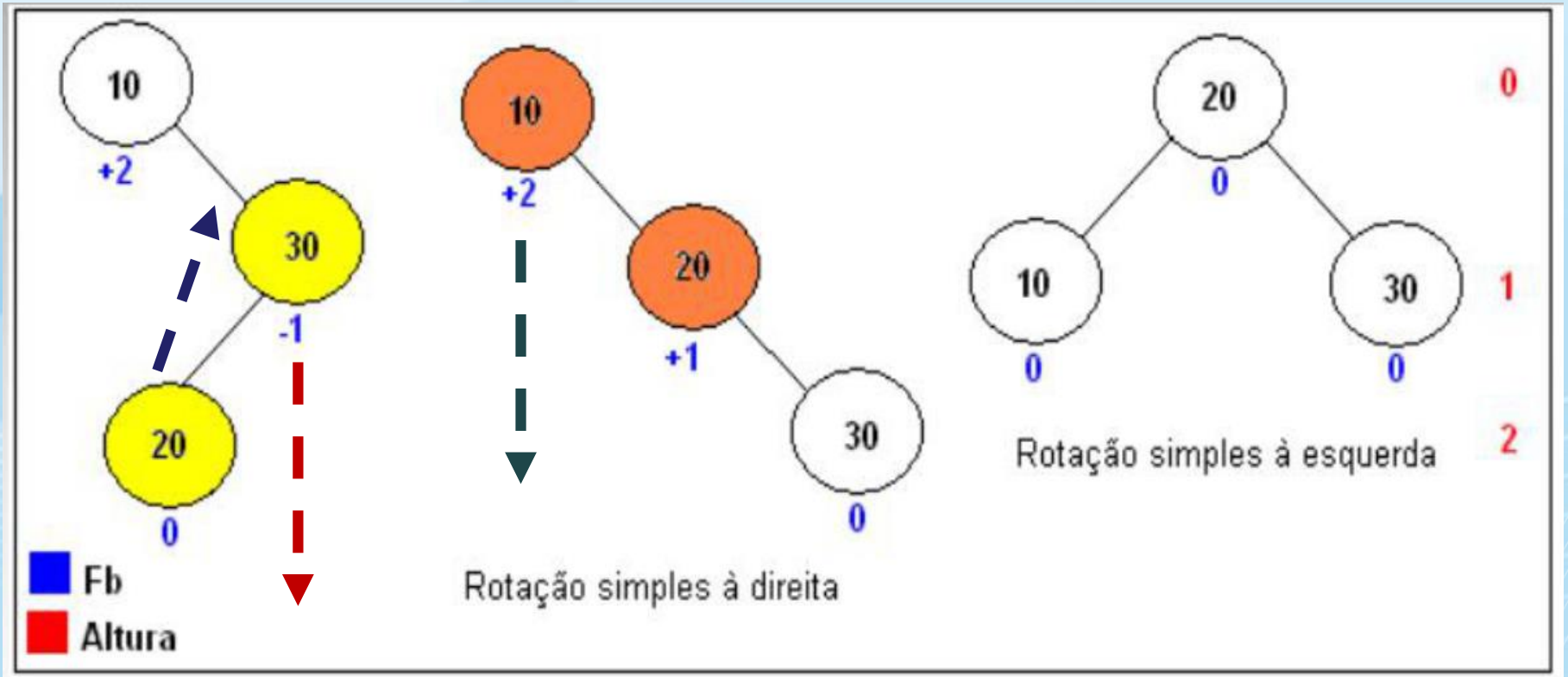


# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## ROTAÇÃO:

## Rotação dupla a esquerda



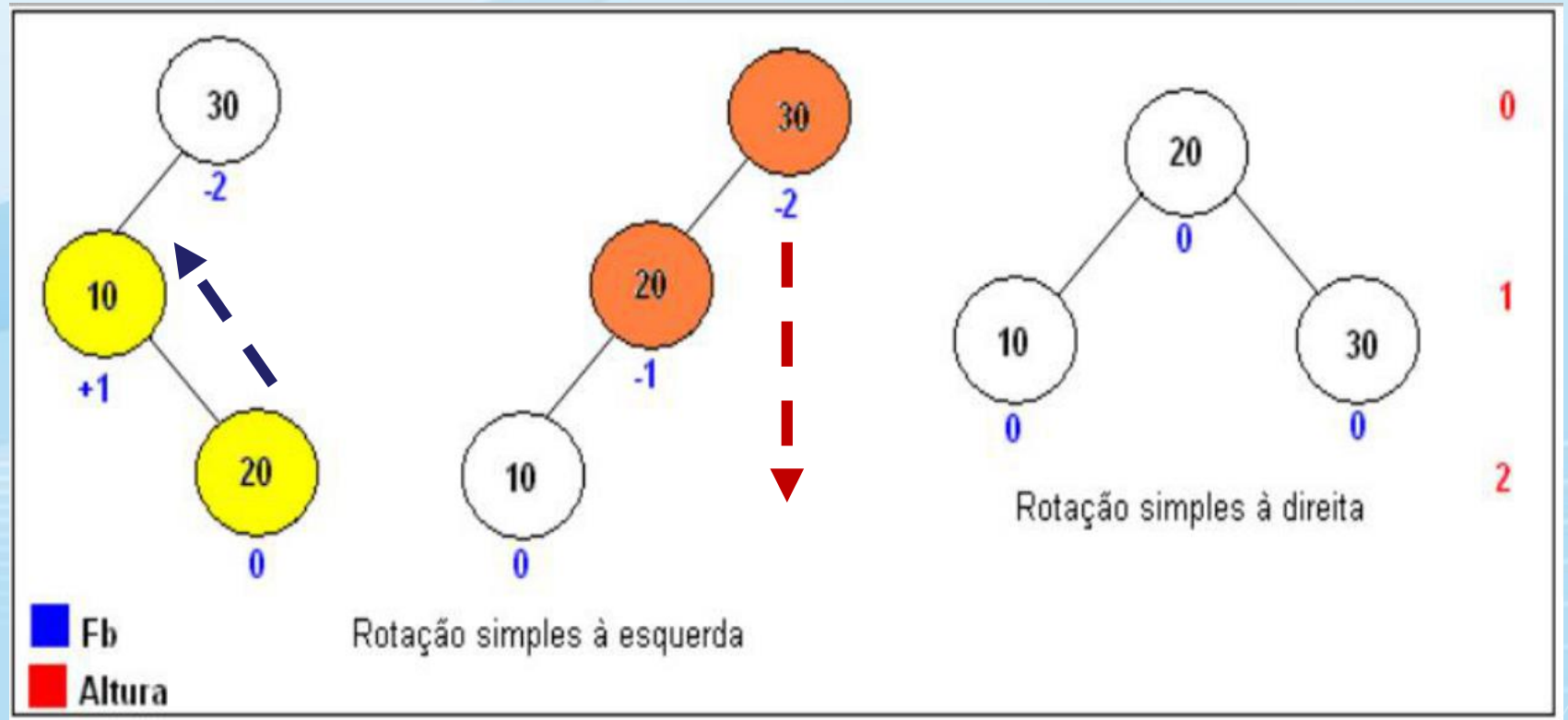

(rotação simples à direita + rotação simples à esquerda)

# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## ROTAÇÃO:

## Rotação dupla a direita




(rotação simples à esquerda + rotação simples à direita)

# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## Rotação:

**Dicas**:

a) Para identificar quando uma rotação é **simples** ou **dupla** deve-se observar os sinais do Fb:

- Sinal for igual, a rotação é simples
- Sinal for diferente a rotação é dupla

b) Se **Fb** for positivo (+) a rotação para à esquerda

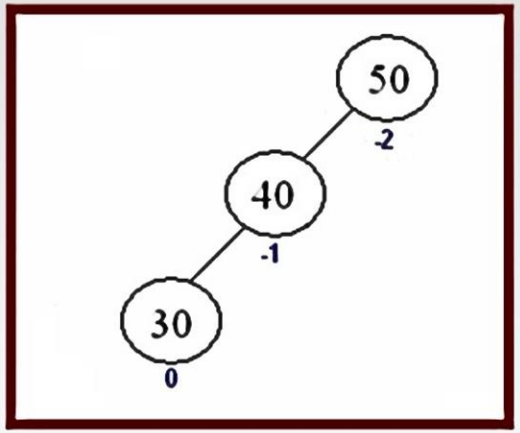
c) Se **Fb** for negativa (-) a rotação para à direita

# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## Caso I: Rotação Simples

- Suponha que inserimos os números 50, 40 e 30 em uma árvore. Obteremos então:



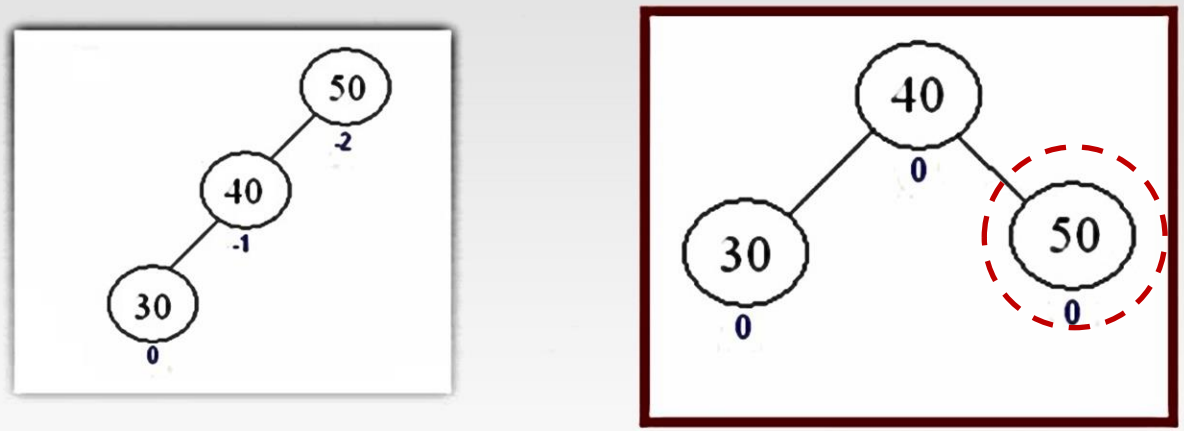

- **A inserção novamente produziu um desbalanceamento.**
- **Neste caso, como os sinais dos FB são os mesmos, significa que precisamos fazer apenas uma ROTAÇÃO SIMPLES à direita no nodo com FB -2.**
- **No caso simétrico (nodo com FB 2) faríamos uma rotação simples à esquerda.**

# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## Caso I: Rotação Simples

- Após a rotação simples teremos:



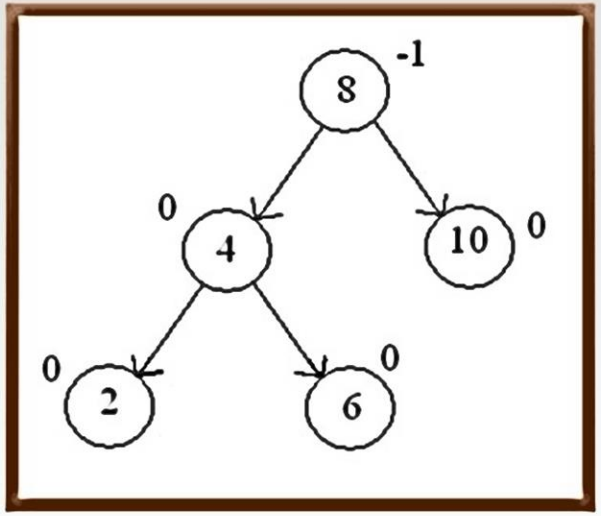

- A árvore está balanceada dentro das propriedades de AVL.

# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## Exemplo:

- Considerando a árvore abaixo:



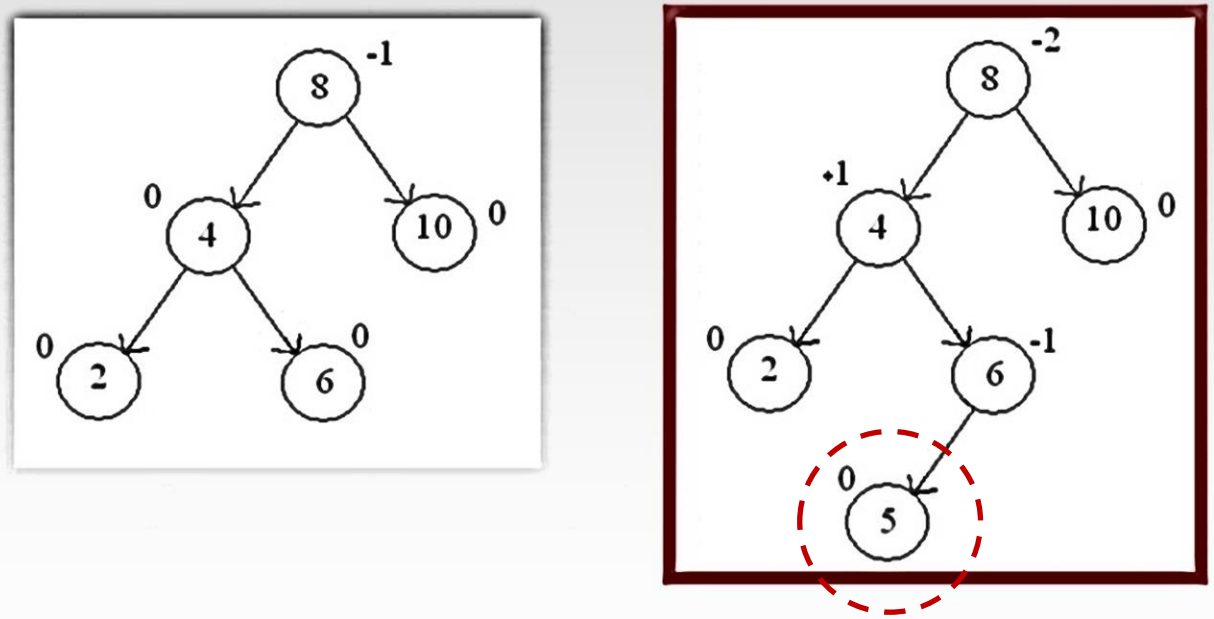

- A árvore está balanceada, como podemos observar pelos Fb de cada nodo.
- São dois os possíveis casos de desbalancemento

# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## Caso II: Rotação Dupla

- Ao inserir o número 5 na árvore teremos a seguinte árvore:

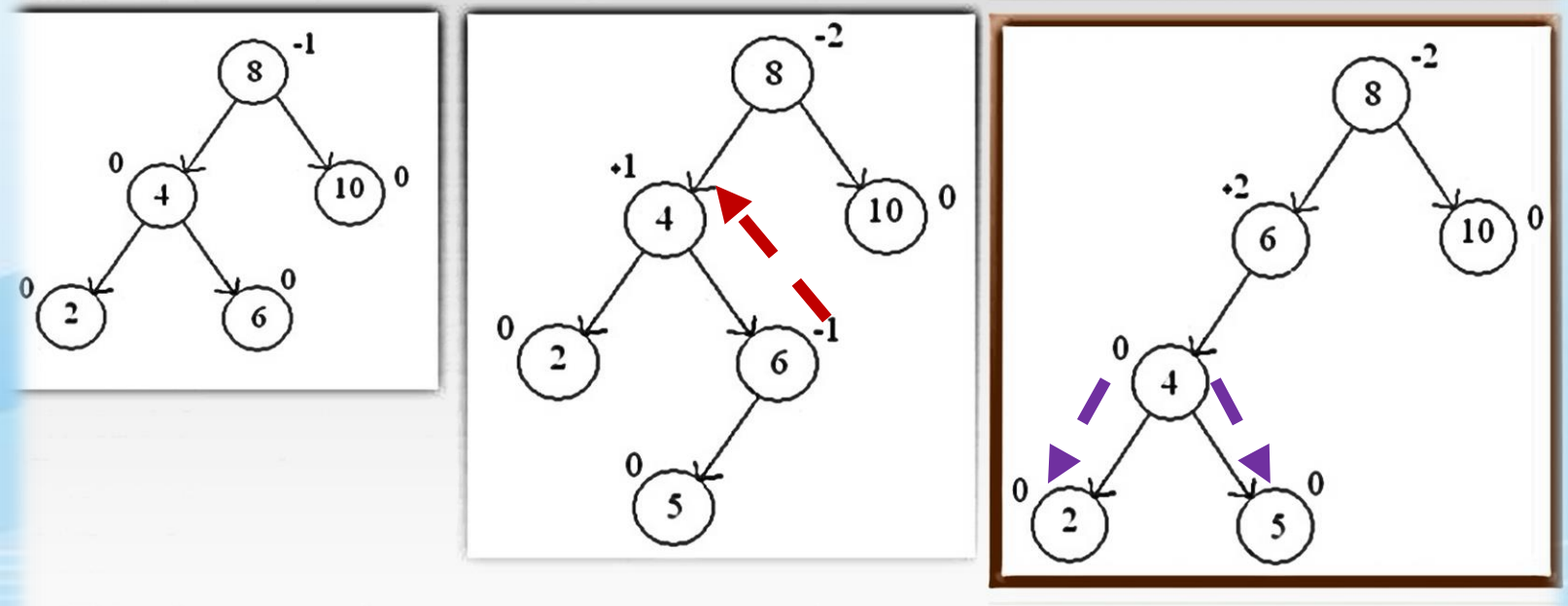



- O nodo 8 fica com o FB -2 e tem um filho com FB +1. Neste caso para manter o balanceamento devemos aplicar duas rotações, também denominada **ROTAÇÃO DUPLA.**
- Primeiro rotaciona-se o nodo com FB 1 para a **esquerda.**

# Aula 12- ALGORITMOS E COMPLEXIDADE

Estruturas de dados dos tipos Árvore Binária e Árvore AVL

## Caso II: Rotação Dupla




- Logo rotaciona-se o nodo que possuía FB -2 na direção oposta, nesse caso a **direita**.

TEMA 4

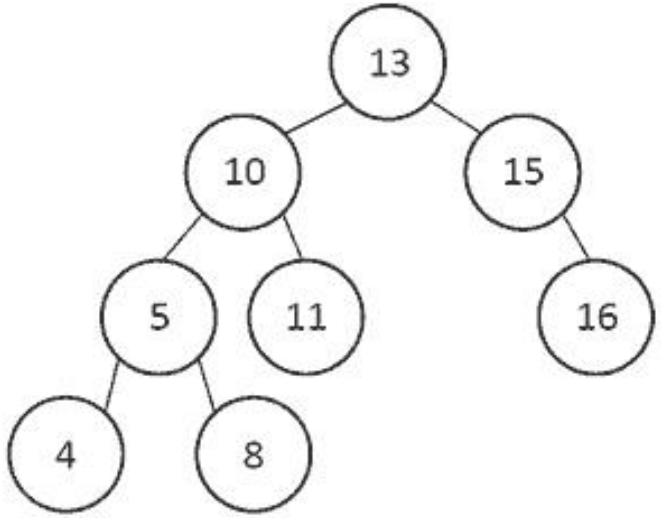
# Aula 12- ALGORITMOS E COMPLEXIDADE

## Estruturas de dados dos tipos Árvore Binária e Árvore AVL

Uma árvore AVL é um tipo de árvore binária balanceada na qual a diferença entre as alturas de suas subárvores da esquerda e da direita não pode ser maior do que 1 para qualquer nó. Após a inserção de um nó em uma AVL, a raiz da subárvore de nível mais baixo no qual o novo nó foi inserido é marcada. Se a altura de seus filhos diferir em mais de uma unidade, é realizada uma rotação simples ou uma rotação dupla para igualar suas alturas.

LAFORE, R. **Data Structures & algorithms in Java**. Indianópolis: Sams Publishing, 2003 (adaptado).

A seguir, é apresentado um exemplo de árvore AVL.




Pelo exposto no texto acima, após a inserção de um nó com valor 3 na árvore AVL exemplificada, qual será a configuração final após a realização de um novo balanceamento?

TEMA 4

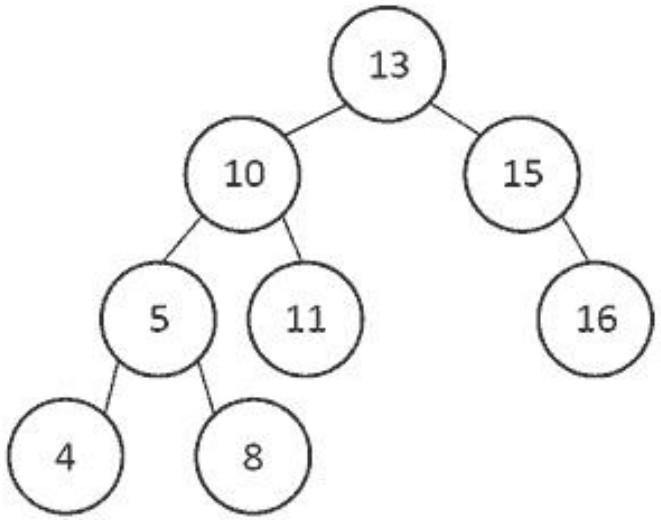
# Aula 12- ALGORITMOS E COMPLEXIDADE

## Estruturas de dados dos tipos Árvore Binária e Árvore AVL

Uma árvore AVL é um tipo de árvore binária balanceada na qual a diferença entre as alturas de suas subárvores da esquerda e da direita não pode ser maior do que 1 para qualquer nó. Após a inserção de um nó em uma AVL, a raiz da subárvore de nível mais baixo no qual o novo nó foi inserido é marcada. Se a altura de seus filhos diferir em mais de uma unidade, é realizada uma rotação simples ou uma rotação dupla para igualar suas alturas.

LAFORE, R. **Data Structures & algorithms in Java**. Indianópolis: Sams Publishing, 2003 (adaptado).

A seguir, é apresentado um exemplo de árvore AVL.




Foi a primeira estrutura de dados a oferecer operações de inserção, remoção e busca em tempo logaritmo ou seja é um **algoritmo muito rápido** .

Em uma árvore degenerada de 10.000 nós, são necessárias 5.000 comparações para efetuar uma busca, já numa árvore AVL, com o mesmo número de nós, essa média baixa para 14 comparações.

A árvore AVL é uma árvore binária de busca e sua estrutura foi construída de forma que a altura da sub-árvore direita é diferente da altura da sub-árvore esquerda de no máximo 1.

Pelo exposto no texto acima, após a inserção de um nó com valor 3 na árvore AVL exemplificada, qual será a configuração final após a realização de um novo balanceamento?

TEMA 4

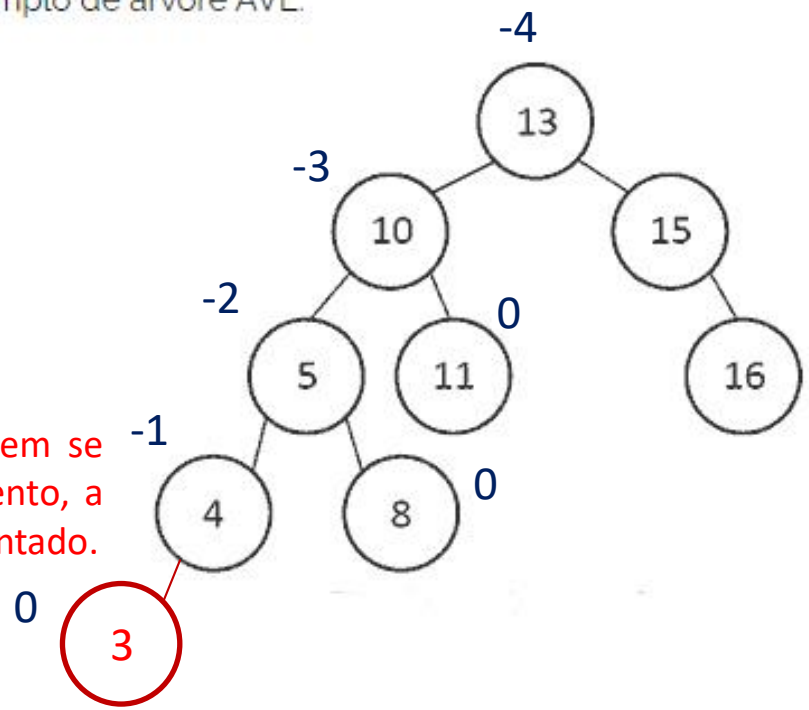
# Aula 12- ALGORITMOS E COMPLEXIDADE

## Estruturas de dados dos tipos Árvore Binária e Árvore AVL

Uma árvore AVL é um tipo de árvore binária balanceada na qual a diferença entre as alturas de suas subárvores da esquerda e da direita não pode ser maior do que 1 para qualquer nó. Após a inserção de um nó em uma AVL, a raiz da subárvore de nível mais baixo no qual o novo nó foi inserido é marcada. Se a altura de seus filhos diferir em mais de uma unidade, é realizada uma rotação simples ou uma rotação dupla para igualar suas alturas.

LAFORE, R. **Data Structures & algorithms in Java**. Indianópolis: Sams Publishing, 2003 (adaptado).

A seguir, é apresentado um exemplo de árvore AVL.




Ao inserir o nó de valor 3, sem se preocupar com o balanceamento, a árvore ficará conforme apresentado.

Pelo exposto no texto acima, após a inserção de um nó com valor 3 na árvore AVL exemplificada, qual será a configuração final após a realização de um novo balanceamento?

TEMA 4

# Aula 12- ALGORITMOS E COMPLEXIDADE

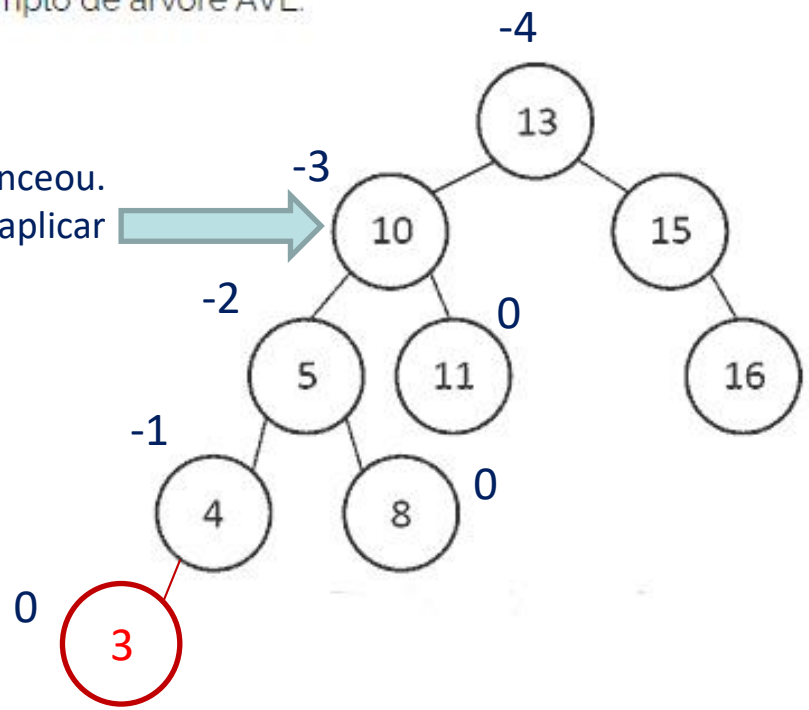
## Estruturas de dados dos tipos Árvore Binária e Árvore AVL

Uma árvore AVL é um tipo de árvore binária balanceada na qual a diferença entre as alturas de suas subárvores da esquerda e da direita não pode ser maior do que 1 para qualquer nó. Após a inserção de um nó em uma AVL, a raiz da subárvore de nível mais baixo no qual o novo nó foi inserido é marcada. Se a altura de seus filhos diferir em mais de uma unidade, é realizada uma rotação simples ou uma rotação dupla para igualar suas alturas.

LAFORE, R. **Data Structures & algorithms in Java**. Indianópolis: Sams Publishing, 2003 (adaptado).

A seguir, é apresentado um exemplo de árvore AVL.

Após a inserção, o nó desbalanceou. Para balanceá-lo, devemos aplicar *uma rotação simples a direita*.




Pelo exposto no texto acima, após a inserção de um nó com valor 3 na árvore AVL exemplificada, qual será a configuração final após a realização de um novo balanceamento?

TEMA 4

# Aula 12- ALGORITMOS E COMPLEXIDADE

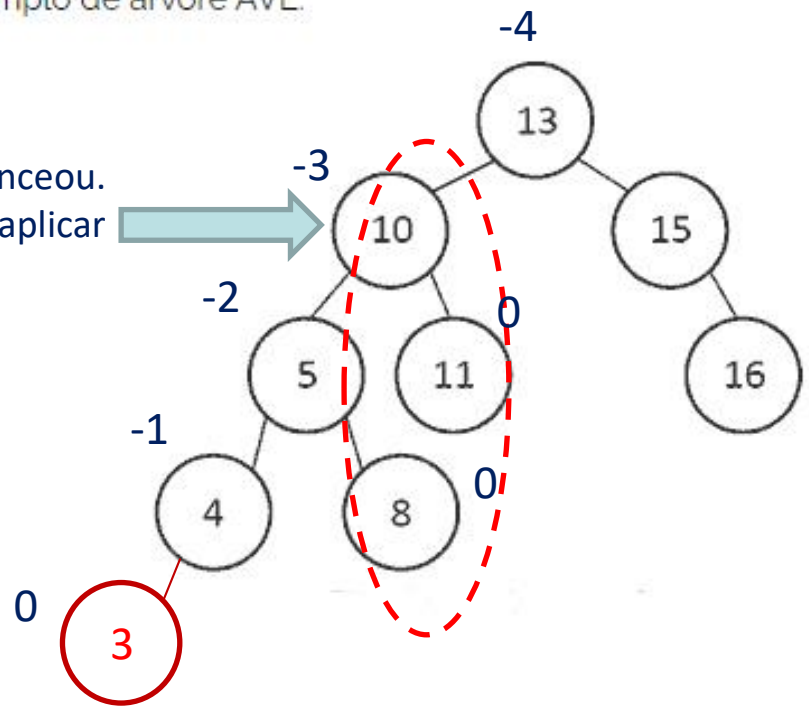
## Estruturas de dados dos tipos Árvore Binária e Árvore AVL

Uma árvore AVL é um tipo de árvore binária balanceada na qual a diferença entre as alturas de suas subárvores da esquerda e da direita não pode ser maior do que 1 para qualquer nó. Após a inserção de um nó em uma AVL, a raiz da subárvore de nível mais baixo no qual o novo nó foi inserido é marcada. Se a altura de seus filhos diferir em mais de uma unidade, é realizada uma rotação simples ou uma rotação dupla para igualar suas alturas.

LAFORE, R. **Data Structures & algorithms in Java**. Indianópolis: Sams Publishing, 2003 (adaptado).

A seguir, é apresentado um exemplo de árvore AVL.

Após a inserção, o nó desbalanceou. Para balanceá-lo, devemos aplicar *uma rotação simples a direita*.




Pelo exposto no texto acima, após a inserção de um nó com valor 3 na árvore AVL exemplificada, qual será a configuração final após a realização de um novo balanceamento?

TEMA 4

# Aula 12- ALGORITMOS E COMPLEXIDADE

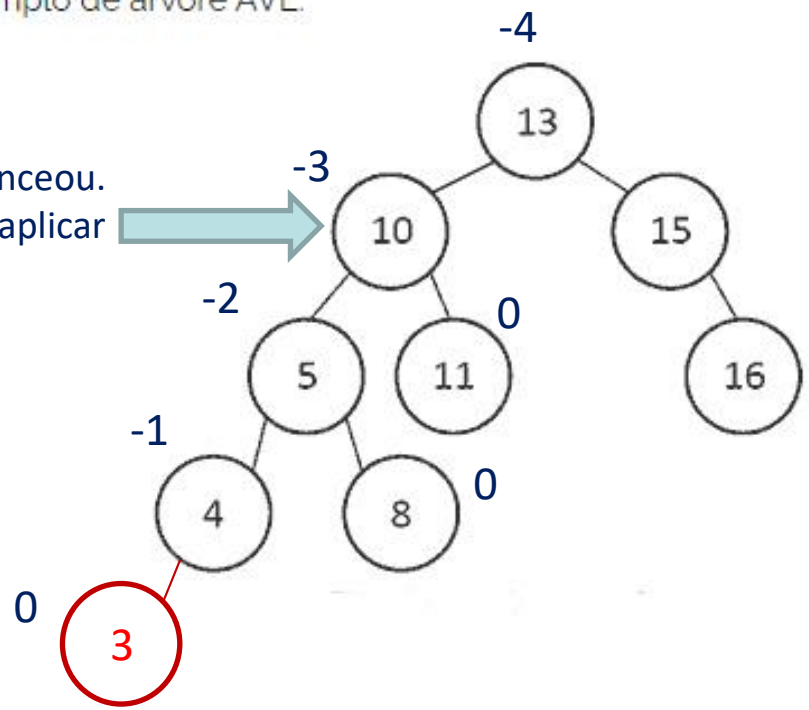
## Estruturas de dados dos tipos Árvore Binária e Árvore AVL

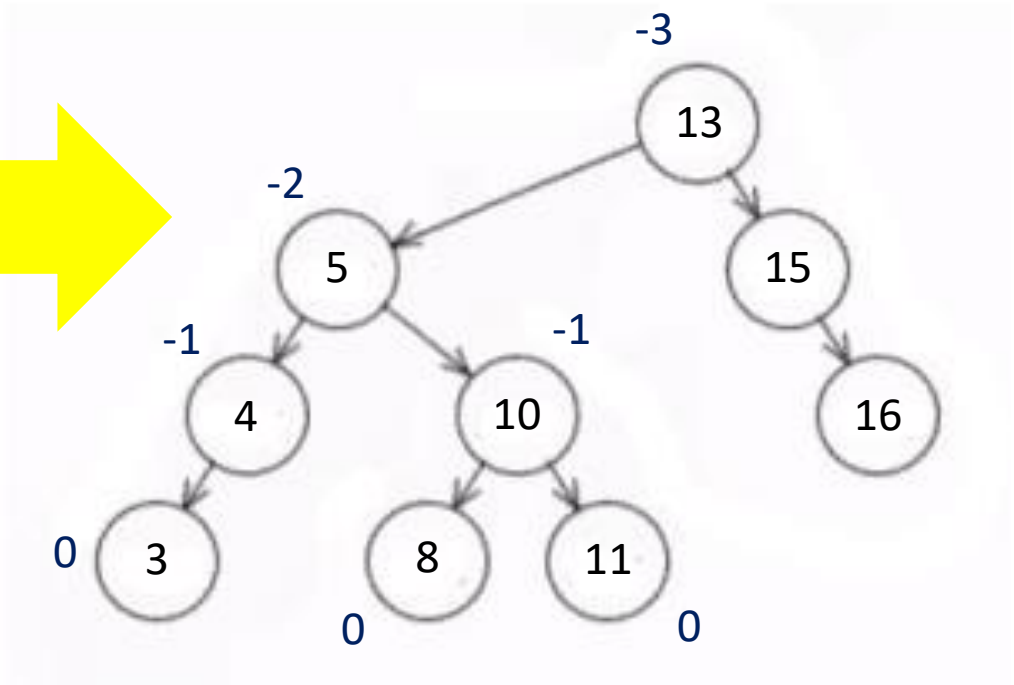
Uma árvore AVL é um tipo de árvore binária balanceada na qual a diferença entre as alturas de suas subárvores da esquerda e da direita não pode ser maior do que 1 para qualquer nó. Após a inserção de um nó em uma AVL, a raiz da subárvore de nível mais baixo no qual o novo nó foi inserido é marcada. Se a altura de seus filhos diferir em mais de uma unidade, é realizada uma rotação simples ou uma rotação dupla para igualar suas alturas.

LAFORE, R. **Data Structures & algorithms in Java**. Indianópolis: Sams Publishing, 2003 (adaptado).

A seguir, é apresentado um exemplo de árvore AVL.

Após a inserção, o nó desbalanceou. Para balanceá-lo, devemos aplicar *uma rotação simples a direita*.




Pelo exposto no texto acima, após a inserção de um nó com valor 3 na árvore AVL exemplificada, qual será a configuração final após a realização de um novo balanceamento?

TEMA 4

# EXEMPLO

```c
#include <stdio.h>
#include <stdLib.h>

typedef struct arvore
{
    int info;
    struct arvore *esq,*dir;
};

void insere(struct arvore **inicio, int info)
{
    struct arvore *aux;
    /* Como a funçao é recursiva, em alguem momento receberá um início igual a NULL,
       ou caso a árvore esteja vazia. */
    if (!*inicio)
    {
        if((aux = (struct arvore*) malloc(sizeof(struct arvore))) != NULL)
        {
            aux -> info = info;
            aux -> dir = NULL;
            aux -> esq = NULL;
            *inicio = aux;
        }
        else
            printf("Nao foi possivel alocar memoria");
    }
    else
    {
        /* Se o info atual for MAIOR que o valor a ser inserido, então esse
           valor será inserido do lado ESQUERDO. */
        if ((*inicio)->info > info)
        insere(&((*inicio)->esq), info);
    else
        /* Se o info atual for MENOR que o valor a ser inserido, então esse
                valor será inserido do lado DIREITO. */
        if((*inicio)->info < info)
            insere(&((*inicio)->dir), info);
```

# EXEMPLO (CONTINUAÇÃO...)

```
        if((*inicio)->info < info)
            insere(&((*inicio)->dir), info);
        /* Caso ele seja igual, isso significa que já pertece a árvore.*/
        else
        printf("%d ja pertence a arvore \n", info);
    }
}

int main()
{
    struct arvore *oi = NULL;
    insere(&oi, 3);
    insere(&oi, 5);
    insere(&oi, 1);
    insere(&oi, 3);

    system("pause");
}
```

## AO COMPILAR E EXECUTAR APARECERÁ NA TELA:

```
3 ja pertence a arvore
Pressione qualquer tecla para continuar. . .
```