



Técnicas de Programação II

Prof. Mario Henrique A. C. Adaniya
Centro Universitário Filadélfia
Ciência da Computação/Sistema de Informação



+ ARVORES



INTRODUÇÃO

Os tipos abstratos de dados estudados foram:

- Listas Simplesmente e Duplamente Encadeadas;
- Listas Circular e Duplamente Circular;
- Fila e Pilha;

A base para estes tipos de dados são as listas lineares, sejam elas estáticas ou dinâmicas.

Embora tais listas apresentem vantagens quanto ao uso, à manipulação e à alocação, ainda possuem problemas:

- 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.



INTRODUÇÃO

Em busca de contornar essas desvantagens, foi proposto o conceito de ÁRVORES.

Apresenta solução eficiente para **inserção, remoção e busca.**



+ ÁRVORES

+ ÁRVORE

As árvores são estruturas de dados adequadas para a representação de hierarquias. A forma mais natural para definirmos uma estrutura de árvore é usando recursividade. Uma árvore é composta por um conjunto de **nós**.



+ ÁRVORE

8

Existe um nó **r**, denominado **raiz**, que contém **zero ou mais sub-árvores**, cujas **raízes** são ligadas diretamente a **r**.

r



+ ÁRVORE

9

COMO É A IMPLEMENTAÇÃO DESSE NÓ?

r

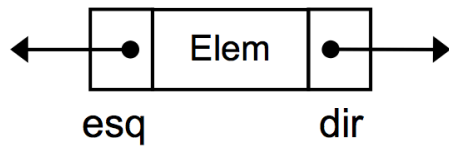


+ ÁRVORE

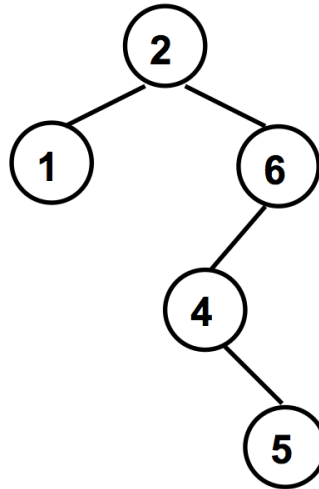
```
1.  public class Node{
2.      int valor;
3.      Node direita;
4.      Node esquerda;
5.
6.      public Node(int valor) {
7.          this.valor = valor;
8.      }
9.  }
```

+ ÁRVORE

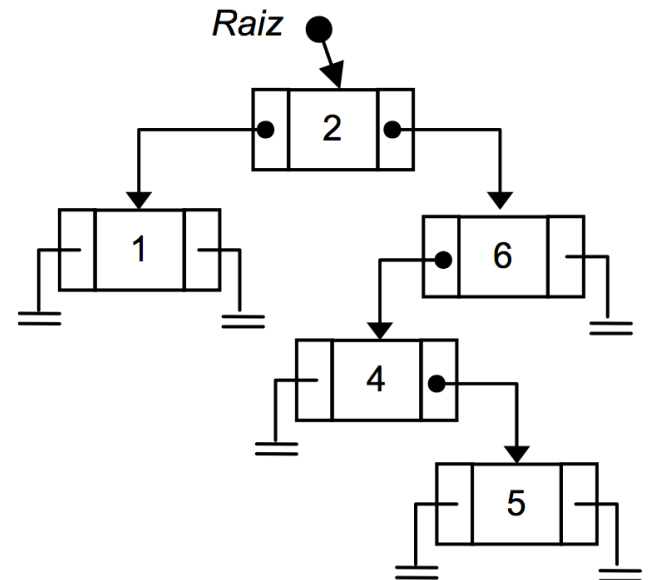
Estrutura do Nó



Representação Gráfica



Representação com nós



Existem 3 formas de representação:

- Por parênteses aninhados;
- Diagrama de inclusão ou Diagramas de Venn;
- Representação hierárquica.



REPRESENTAÇÃO

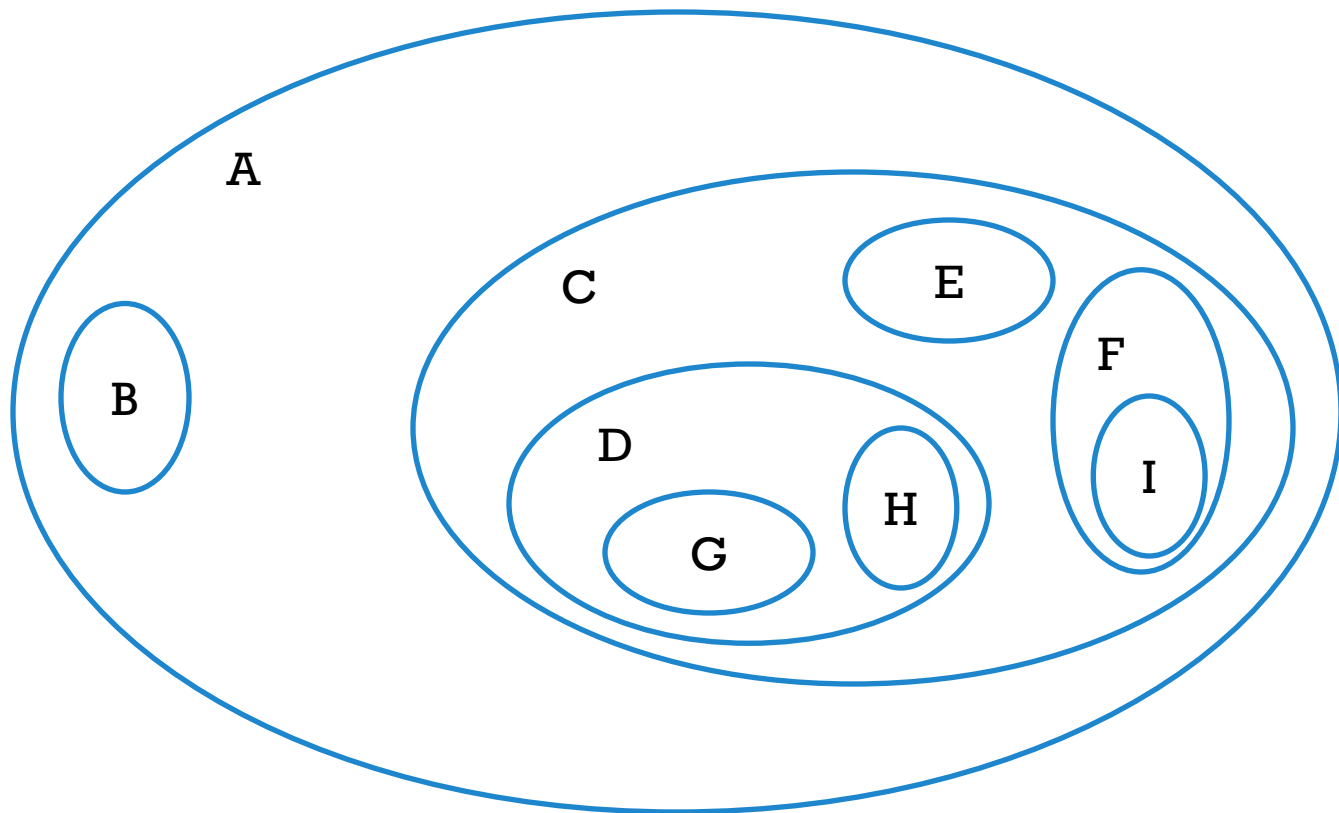
Por parênteses aninhados

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



REPRESENTAÇÃO

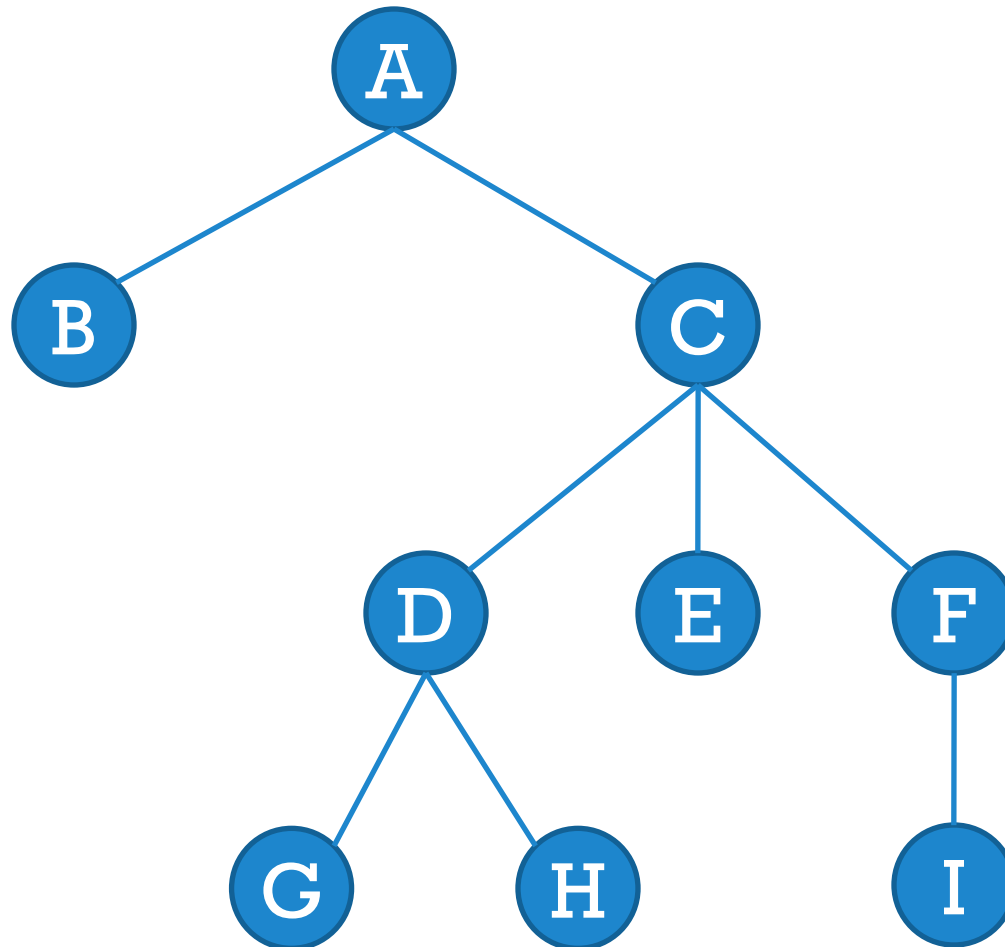
Diagrama de inclusão/Diagramas de Venn





REPRESENTAÇÃO

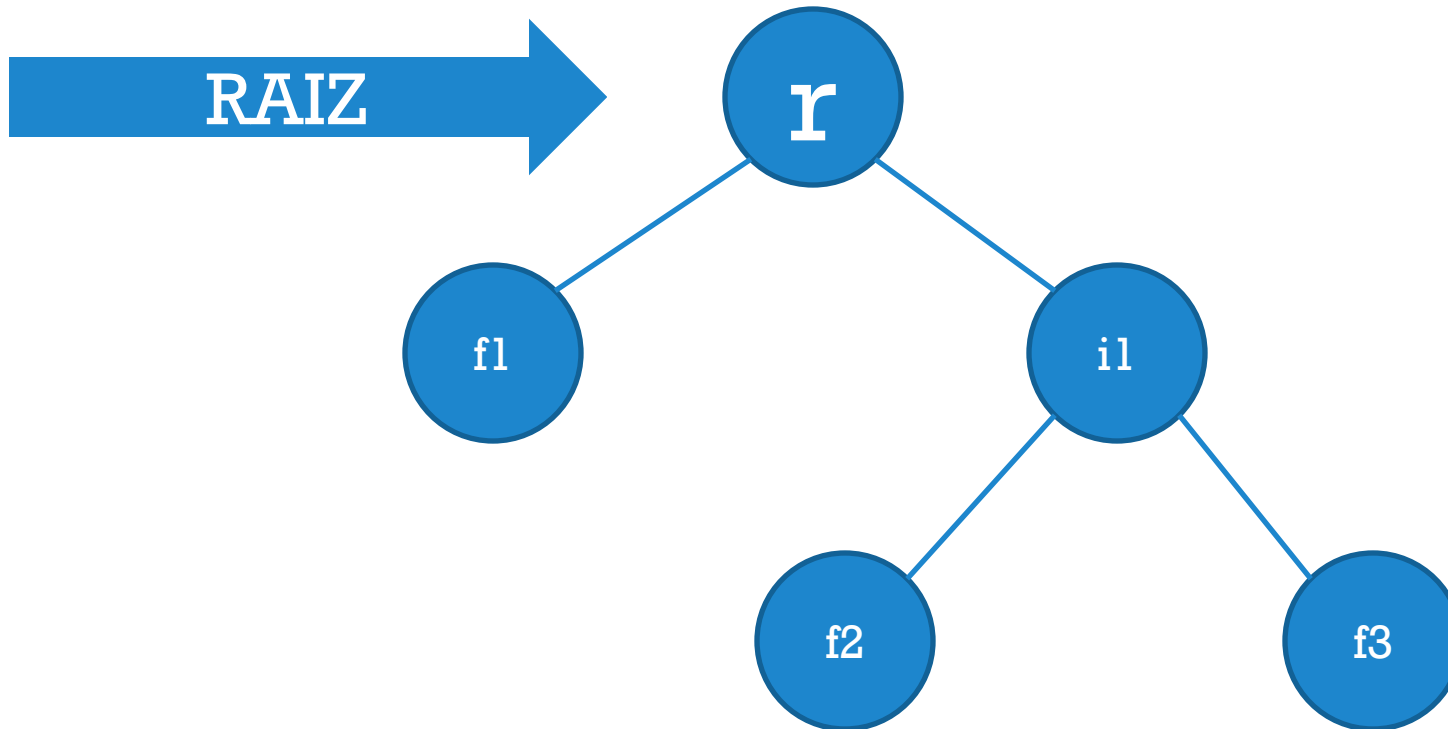
Representação hierárquica.



Esses nós raízes das sub-árvores são ditos filhos do nó pai, r . Nós com filhos são comumente chamados de nós internos e nós que não têm filhos são chamados de folhas, ou nós externos.

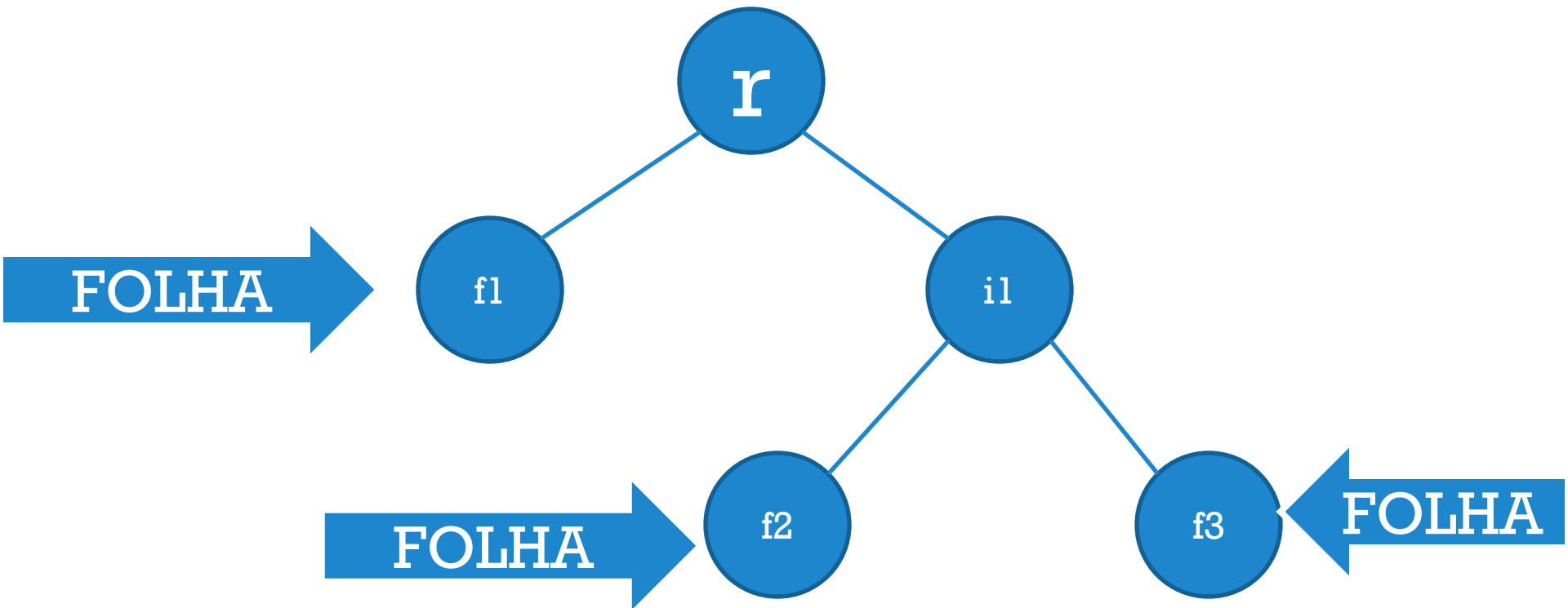
É comum representar as árvores com a raiz para cima e folhas para baixo, ao contrário do que normalmente encontramos no mundo real.

+ ÁRVORES

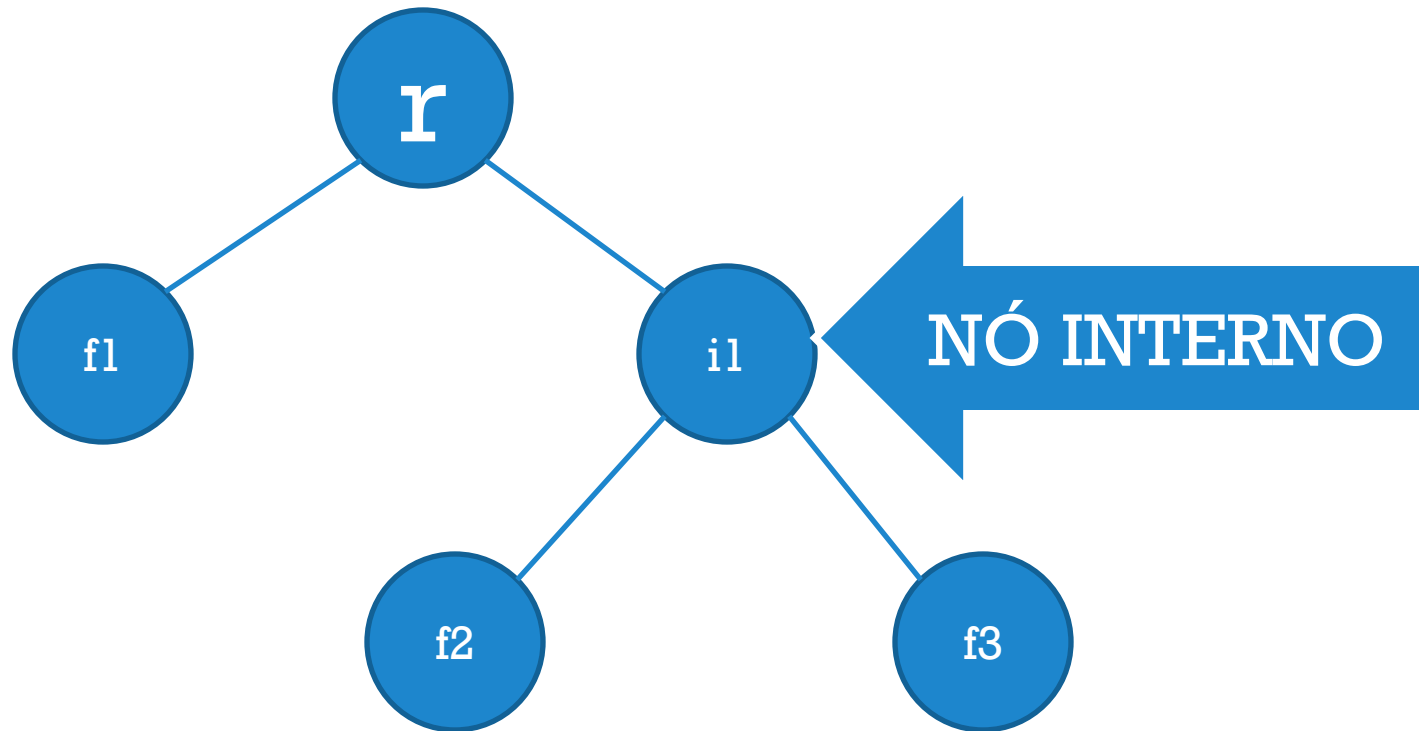


+ ÁRVORES

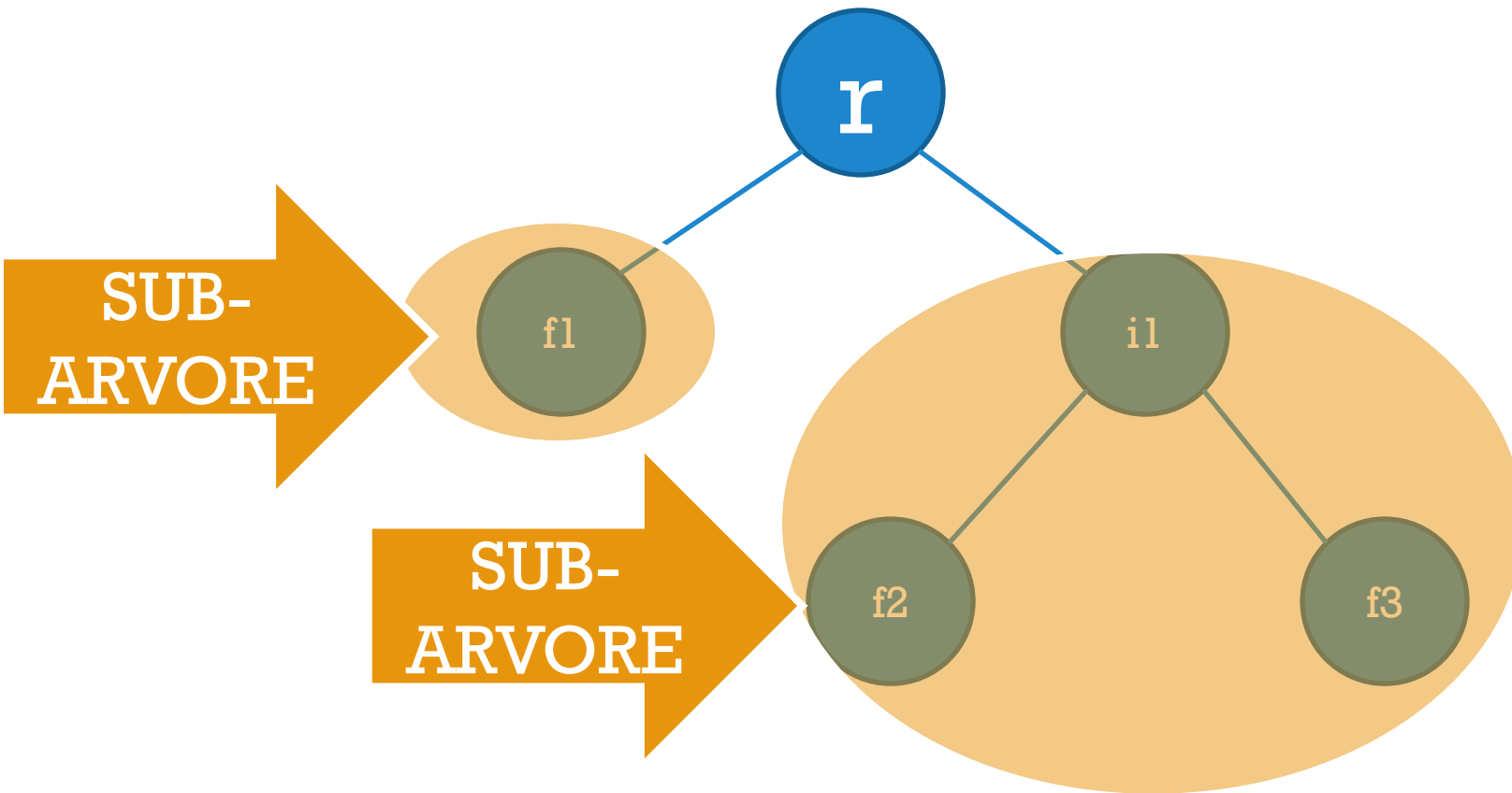
18



+ ÁRVORES



+ ÁRVORES

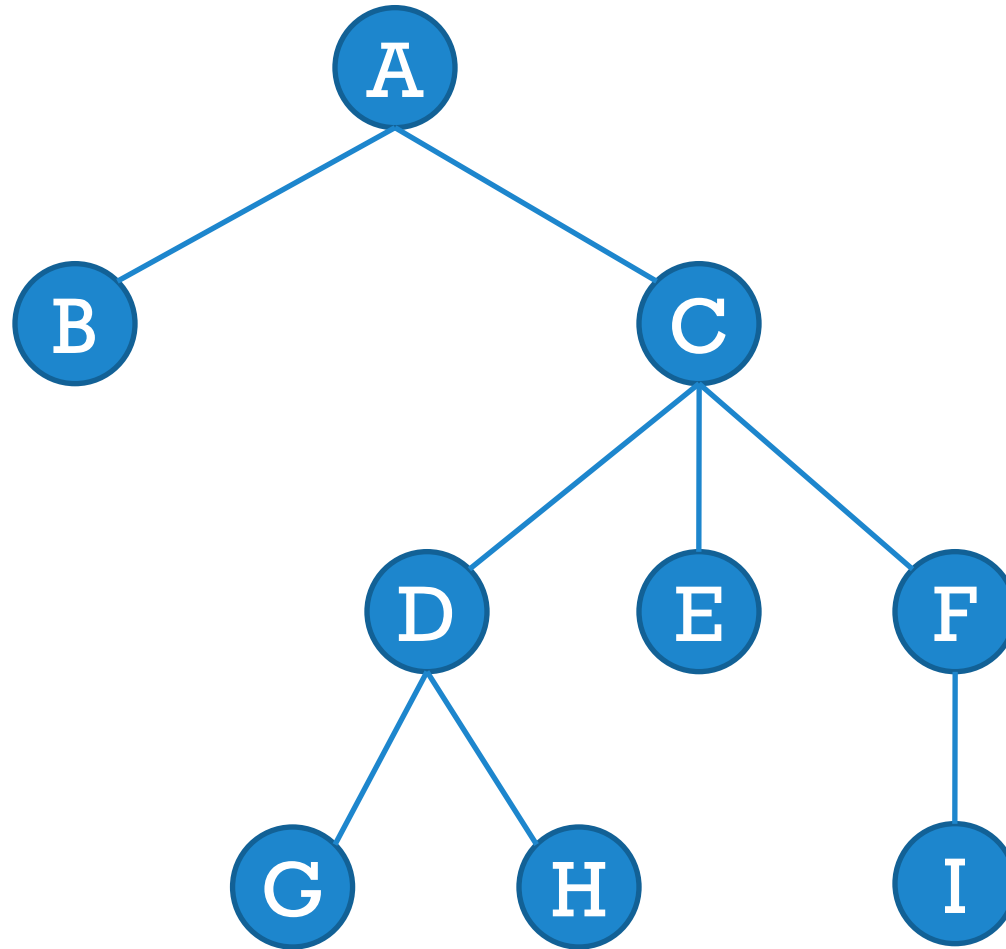




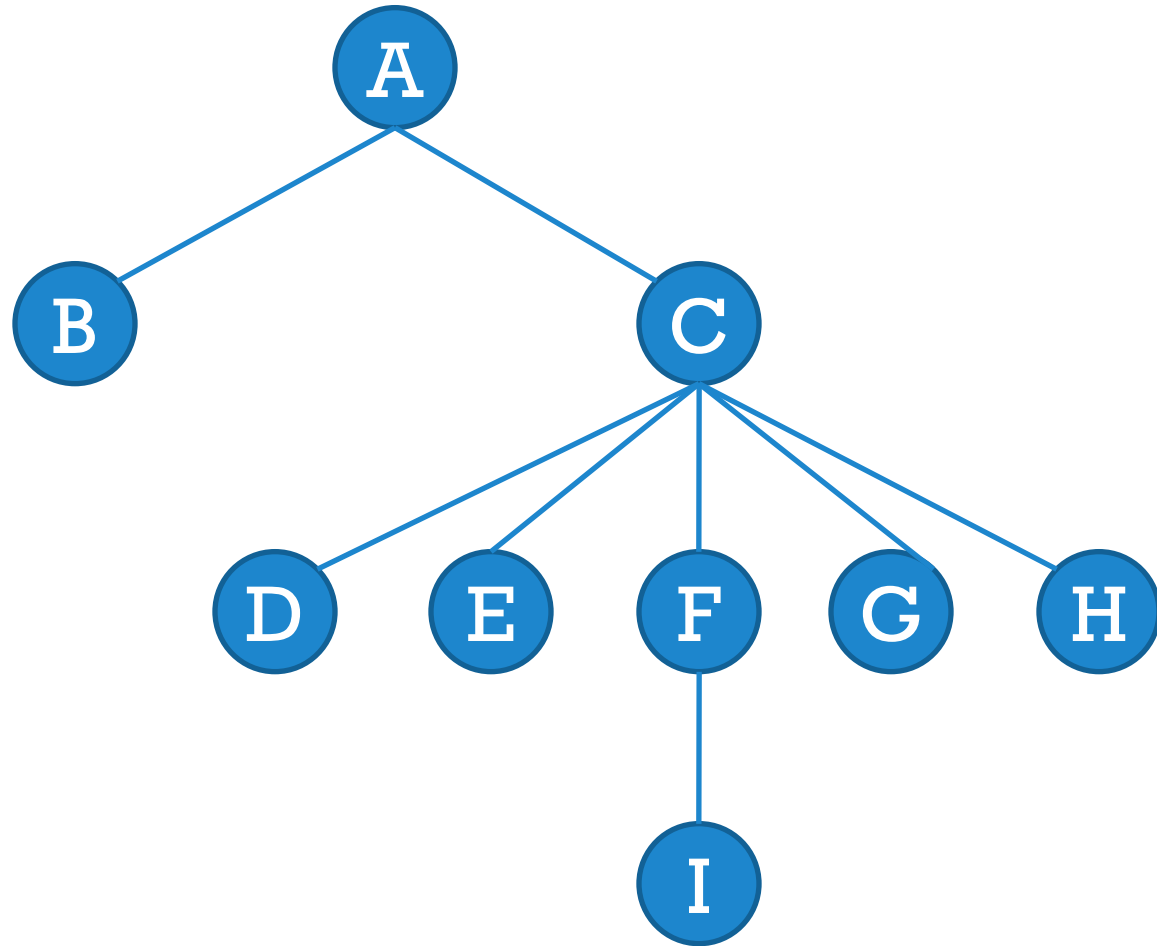
ÁRVORE

O número de filhos permitido por nó e as informações armazenadas em cada nó diferenciam os diversos tipos de árvores existentes.

+ ÁRVORE



+ ÁRVORE



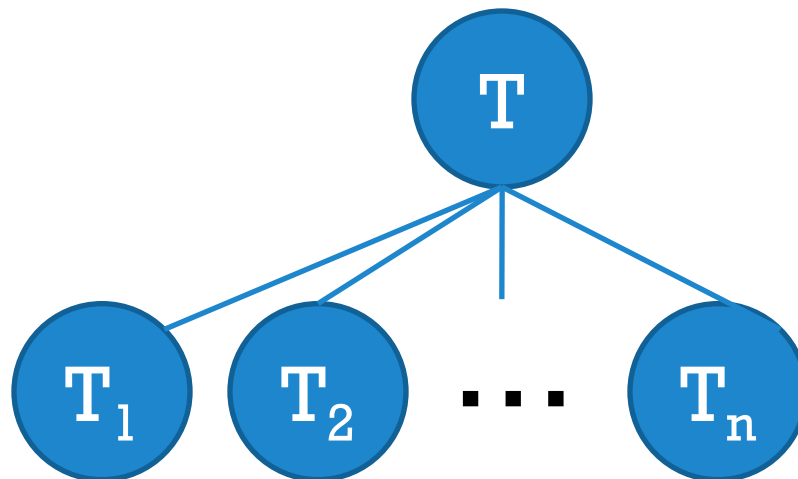
+ ÁRVORE

DEFINIÇÃO

- Uma árvore é um conjunto finito de elementos denominados **nós** ou **vértices** tais que:
 - Se $T = \emptyset$, a árvore é dita vazia;
 - Caso contrário:
 - T contém um nó especial, denominado raiz;
 - os demais nós ou constituem um único conjunto vazio, ou são divididos em $m \geq 1$ conjuntos disjuntos não vazios (T_1, T_2, \dots, T_n) , que são, por sua vez, cada qual uma árvore;

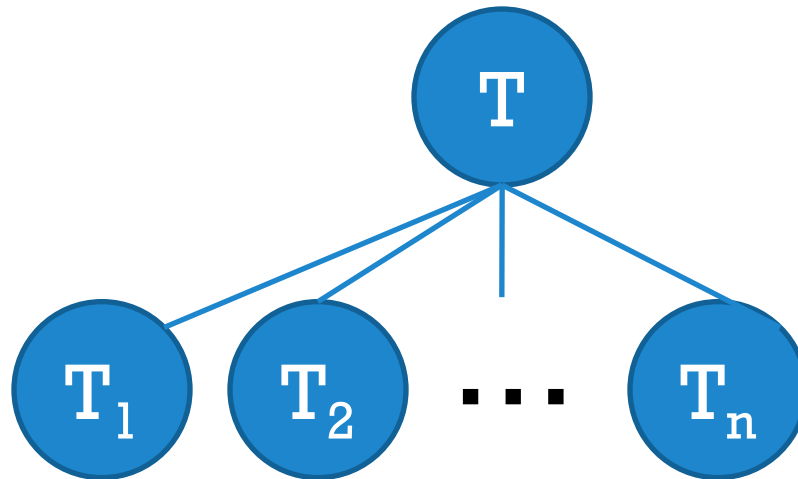
+ ÁRVORE

- T_1, T_2, \dots, T_n são chamadas sub-árvores de T ;
- Um nó sem sub-árvores é denominado nó-folha, ou simplesmente, folha.



+ ÁRVORE

- Se um nó T é a raiz de uma árvore e um nó T_1 é raiz de uma sub-árvore de T , então T é o **PAI** de T_1 e T_1 é o **FILHO** de T .





ÁRVORE TERMINOLOGIA

- **Subárvore:** Cada nó da árvore é a raiz de uma subárvore.
- **Grau:** O número de subárvores de um nó é o grau daquele nó.
- **Folha:** Um nó de grau igual a zero é denominado folha ou nó terminal.

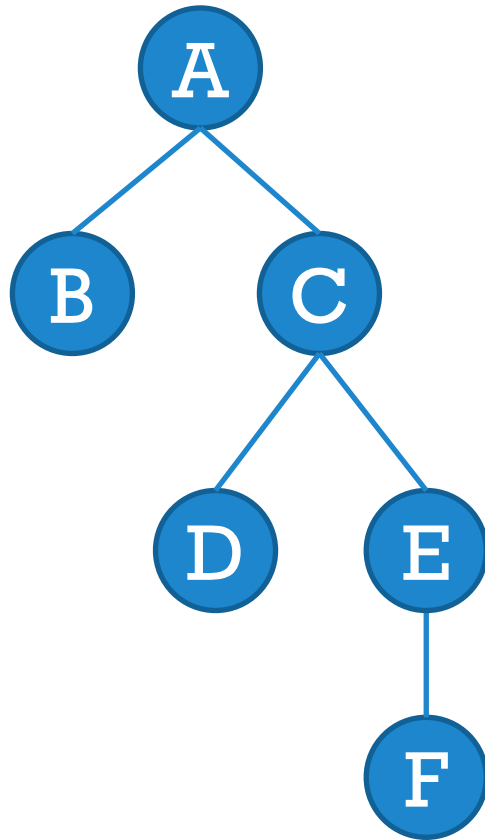
ÁRVORE

TERMINOLOGIA

- **Nível:** O nível do nó é definido da seguinte forma: a raiz da árvore tem nível 0, enquanto o nível dos demais nós é igual ao número de linhas que o liga à raiz, i.e., é o comprimento do caminho que vai da raiz até este nó.
- **Altura:** É definida como sendo o nível mais alto da árvore.



ÁRVORE TERMINOLOGIA



Nó	Grau	Nível
A	2	0
B	0	1
C	2	1
D	0	2
E	1	2
F	0	3

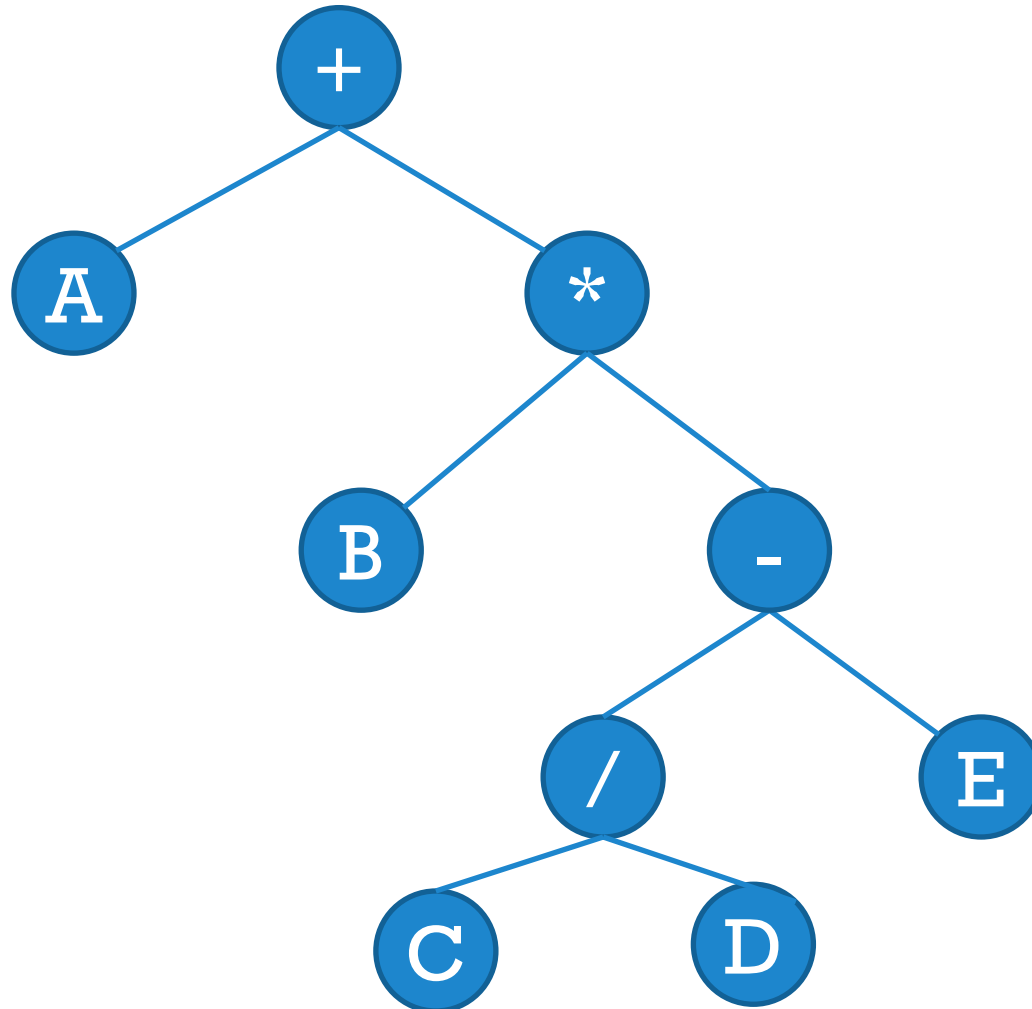
+ ÁRVORE

- Considere a seguinte expressão aritmética:

- $(a + (b * (c / d) - e))$

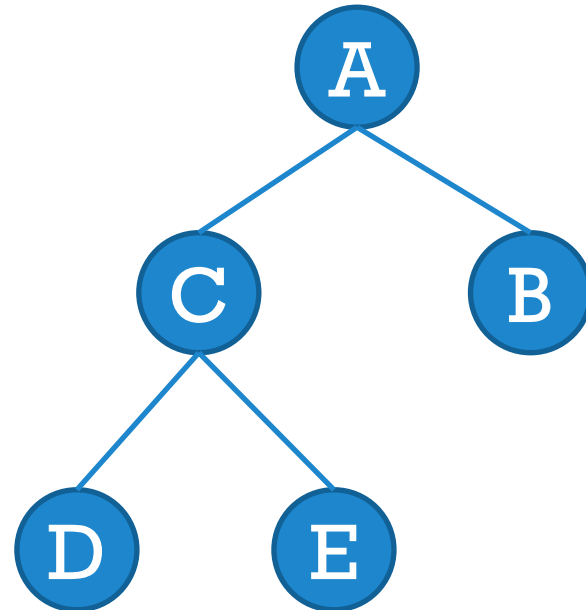
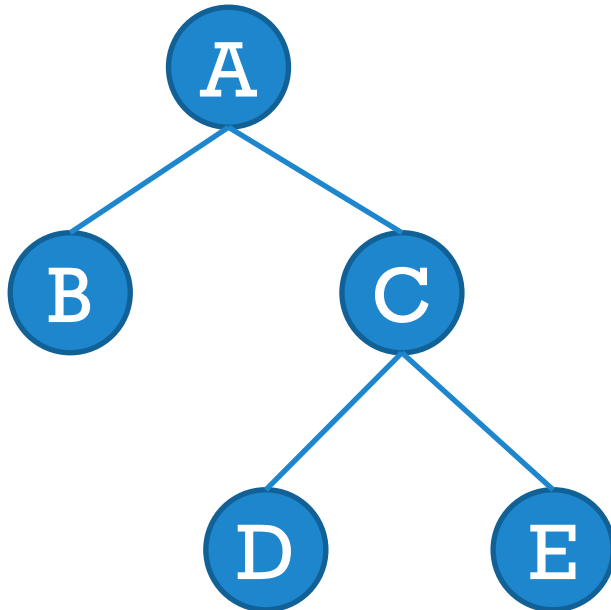
+ ÁRVORE

$(a + (b * (c / d) - e))$



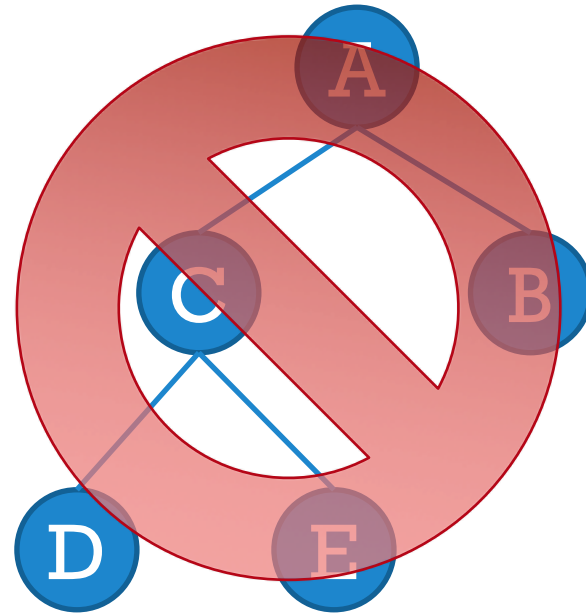
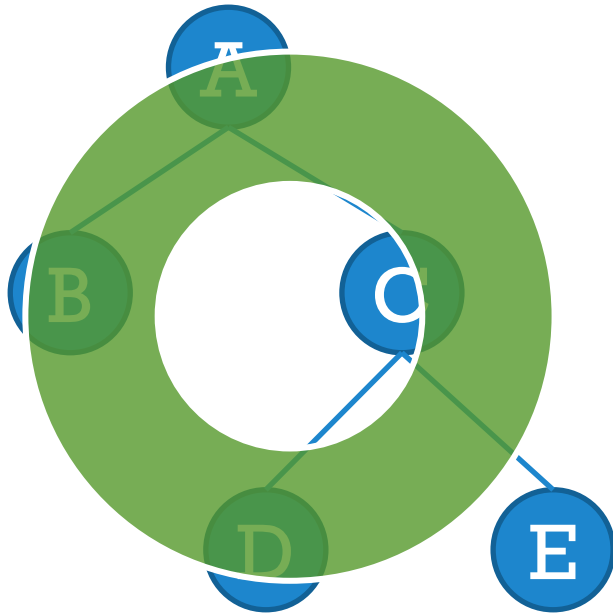
+ ÁRVORE ORDENADA

- É aquela na qual os filhos de cada nó estão ordenados. Assume-se a ordenação da esquerda para direita.



+ ÁRVORE ORDENADA

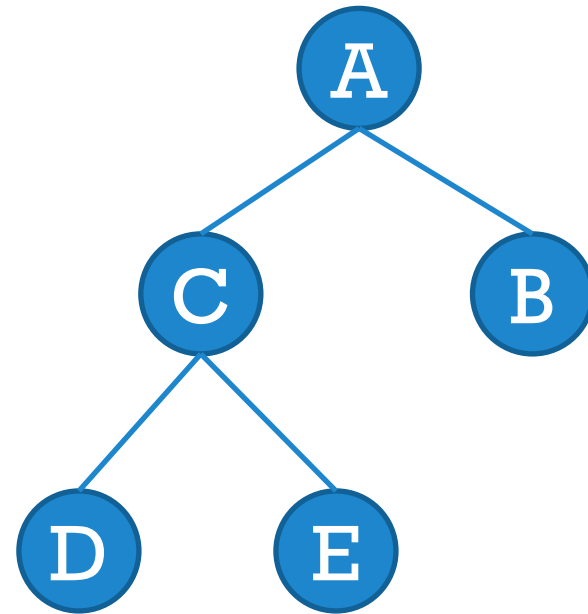
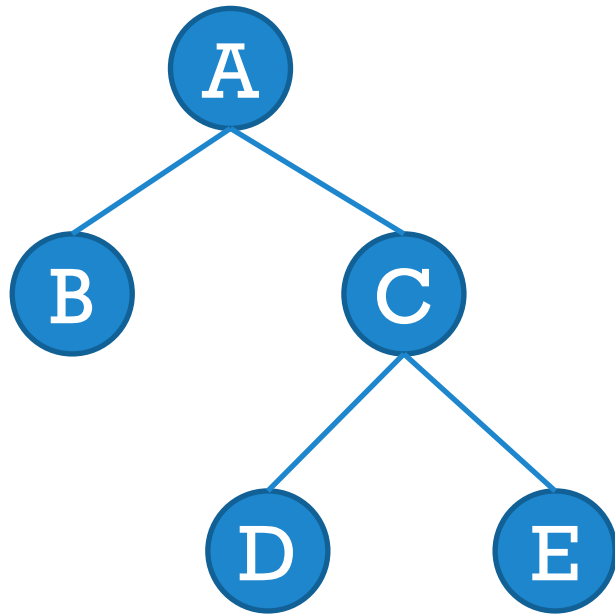
- É aquela na qual os filhos de cada nó estão ordenados. Assume-se a ordenação da esquerda para direita.



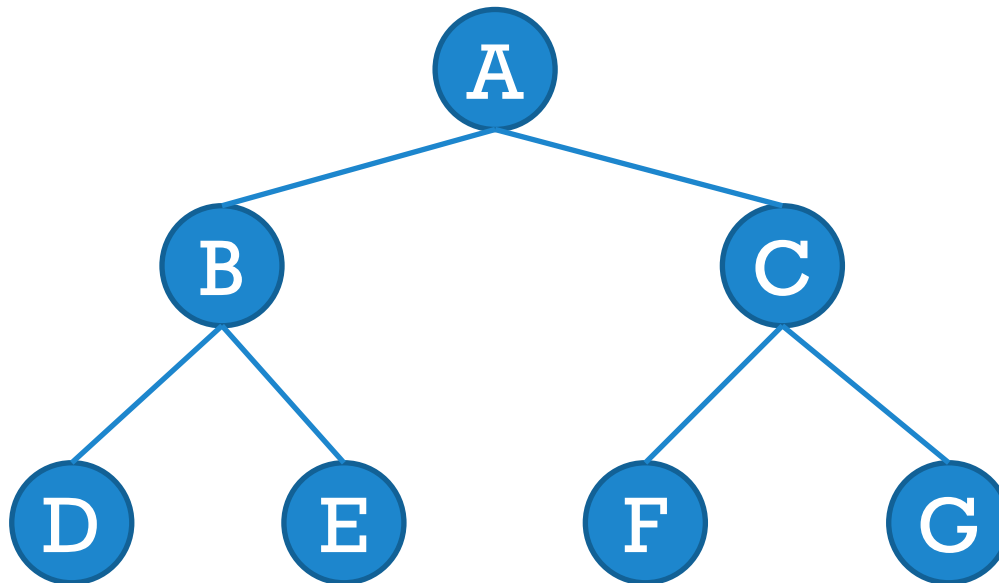
+ ÁRVORE ISOMORFAS

- Duas árvores não ordenadas são isomórfas quando puderem se tornar coincidentes através de uma permutação na ordem das subárvores de seus nós.

+ ÁRVORE ISOMORFAS

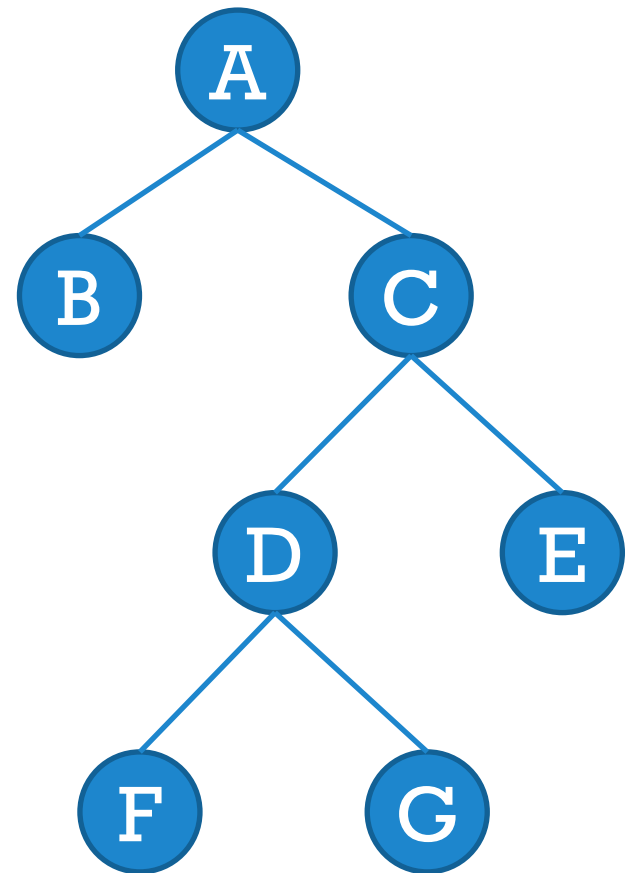
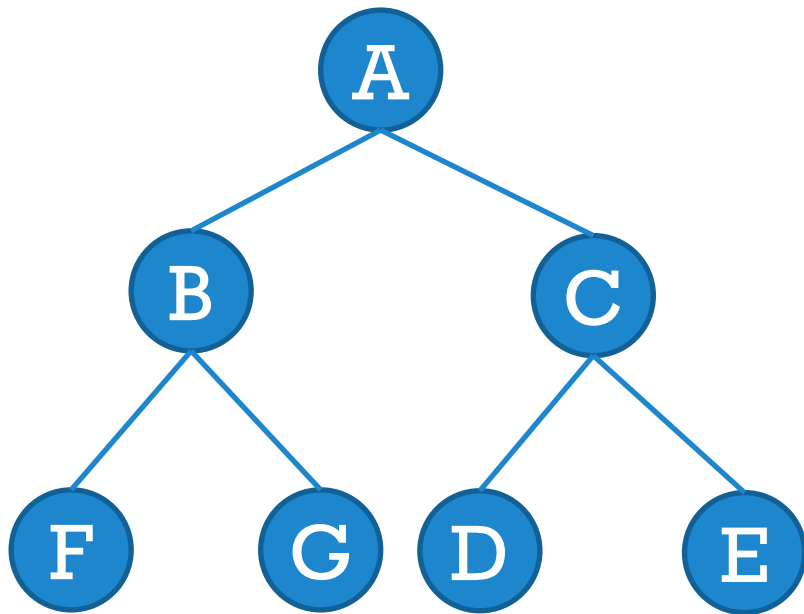


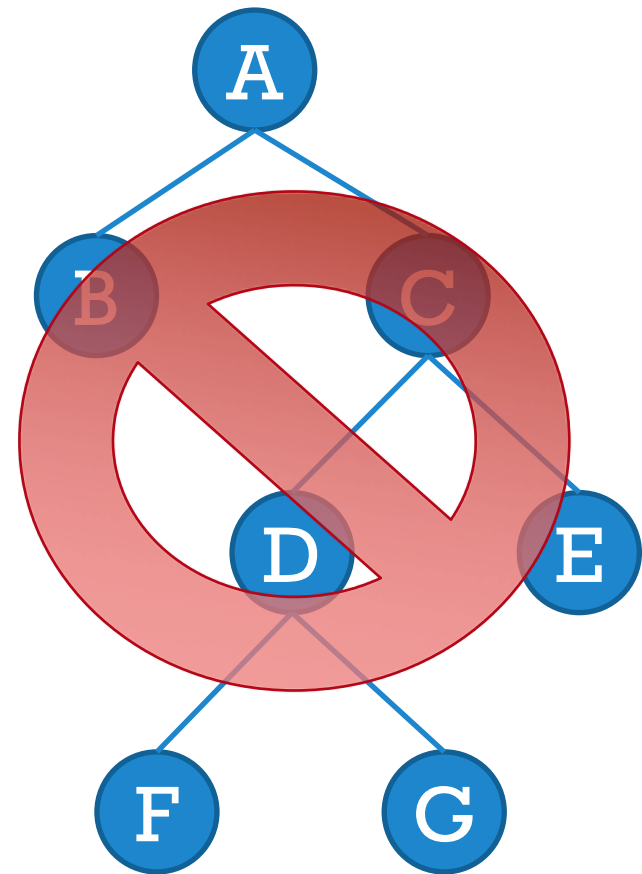
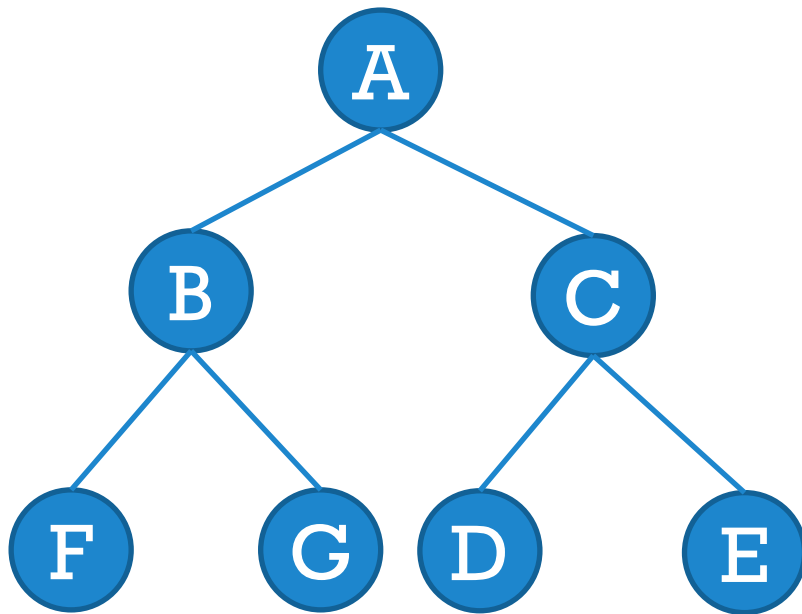
- Uma árvore de grau d é uma árvore cheia se possui o número máximo de nós, isto é, todos os nós tem número máximo de filhos exceto as folhas, e todas as folhas estão na mesma altura.





ÁRVORES BINÁRIAS COMPLETAS





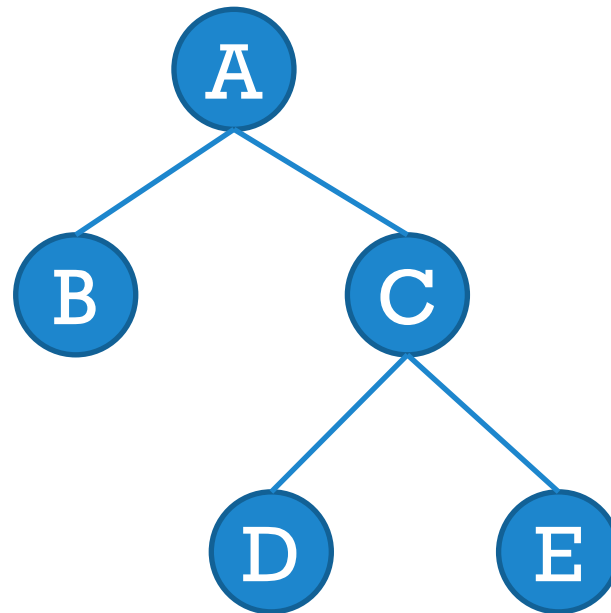
+ ÁRVORES BINÁRIAS

- Uma árvore binária **T** é um conjunto finito de elementos denominados nós ou vértices, tal que:
 - $T = 0$ e a árvore é dita vazia ou
 - existe um nó especial **r**, chamado raiz de **T**, os restantes podem ser divididos em dois subconjuntos disjuntos, **T_{re}** e **T_{rd}**, que são as subárvores esquerda e direita de **r**, respectivamente e as quais, por sua vez, também são árvores binárias.



ÁRVORE BINÁRIA

40

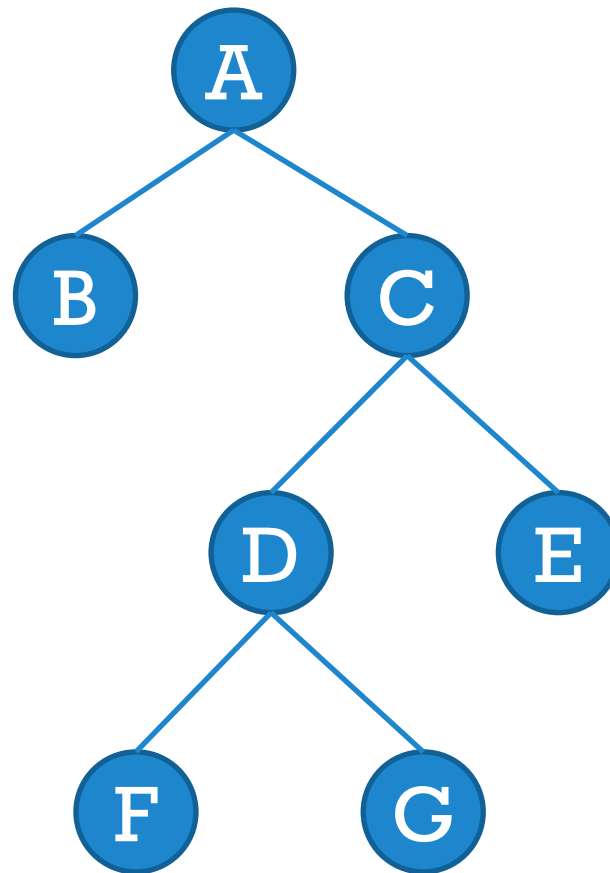


+ ÁRVORES ESTRITAMENTE BINÁRIAS

- Se todo nó que não é folha numa árvore binária tiver sub-árvores esquerda e direita não vazias. Uma árvore estritamente binária com n folhas tem $2n-1$ nós.
- Nós interiores (não folhas) possuem sempre 2 filhos.



ÁRVORES ESTRITAMENTE BINÁRIAS

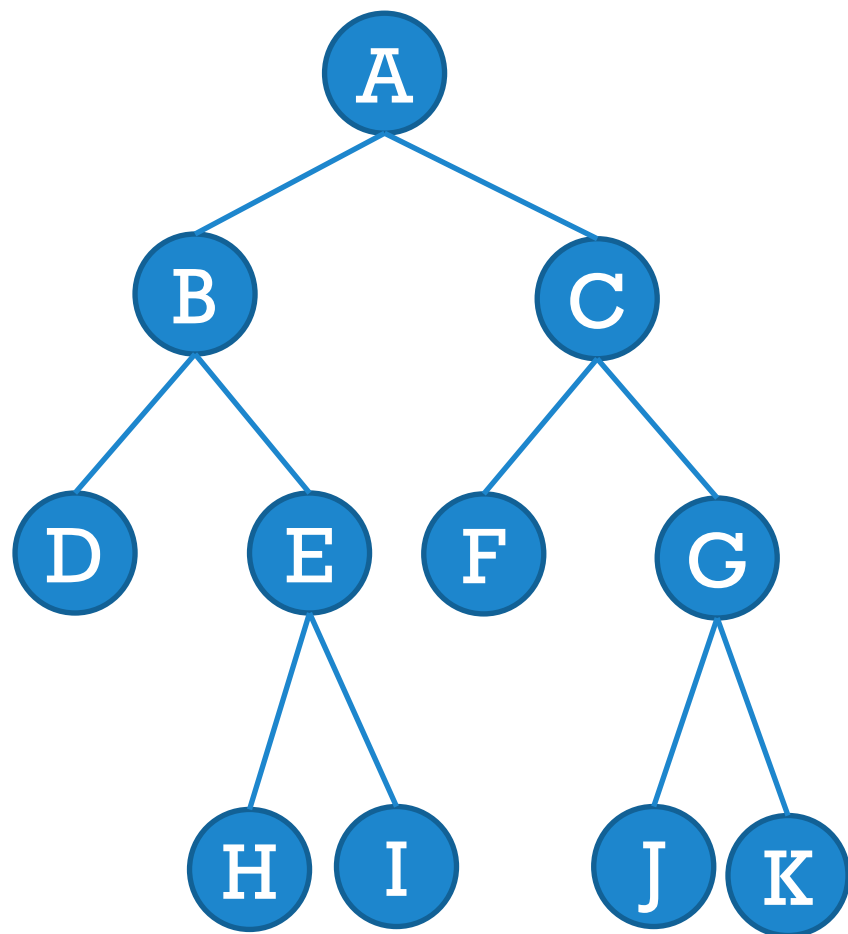
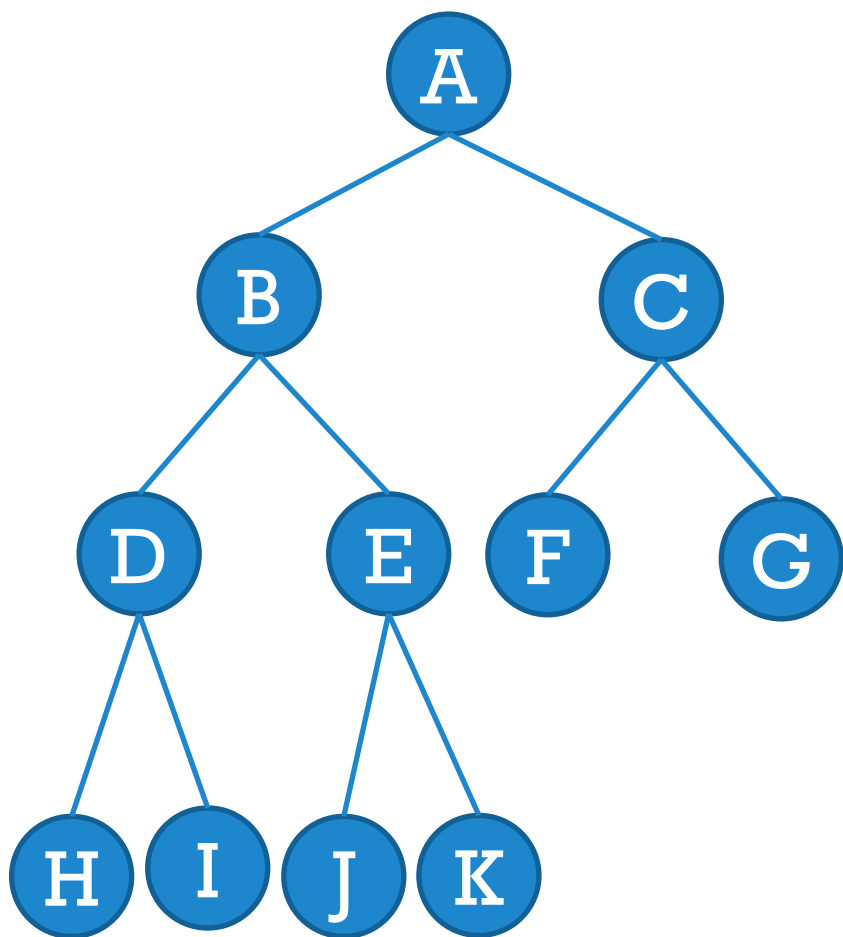


+ ÁRVORES BINÁRIAS QUASE COMPLETA

- É uma Árvore Binária com profundidade k onde:
 - Cada nó folha na árvore está no nível k ou no nível $k-1$ (até o nível $k-1$ ela é completa);
 - Para qualquer nó n da árvore com um descendente direito no nível k , também deve existir o descendente esquerdo correspondente no nível k .
 - Uma árvore binária quase completa pode ter seus nós numerados começando da raiz, de cima para baixo e da esquerda para a direita sem que haja a ausência de nós.



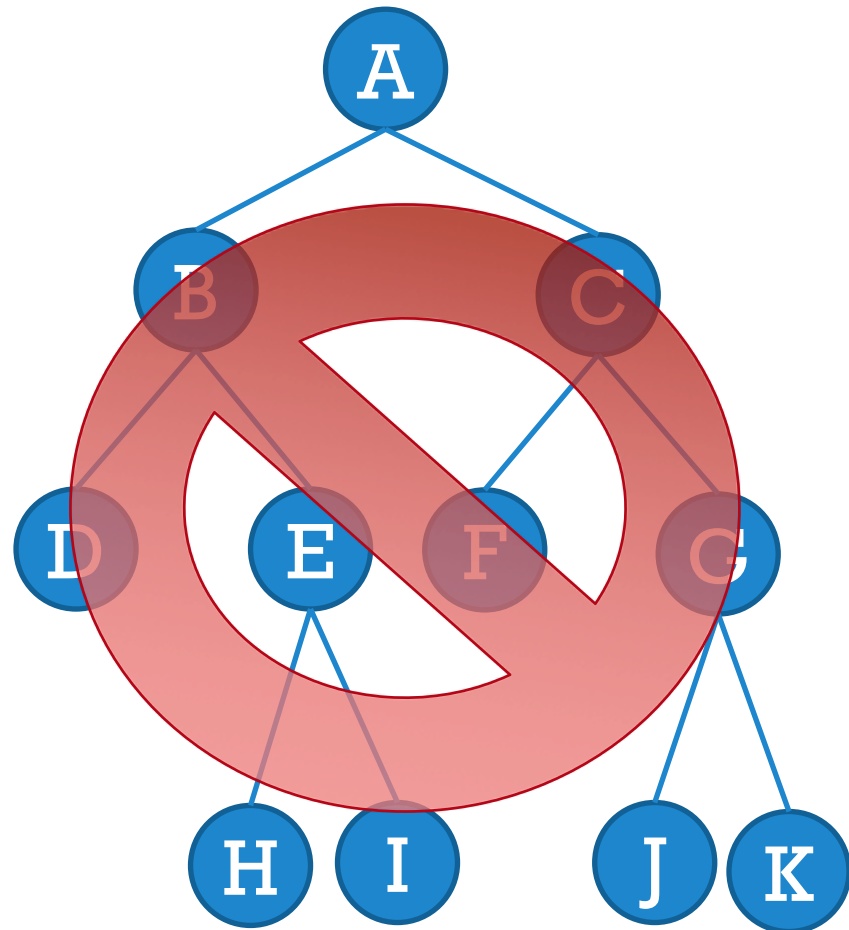
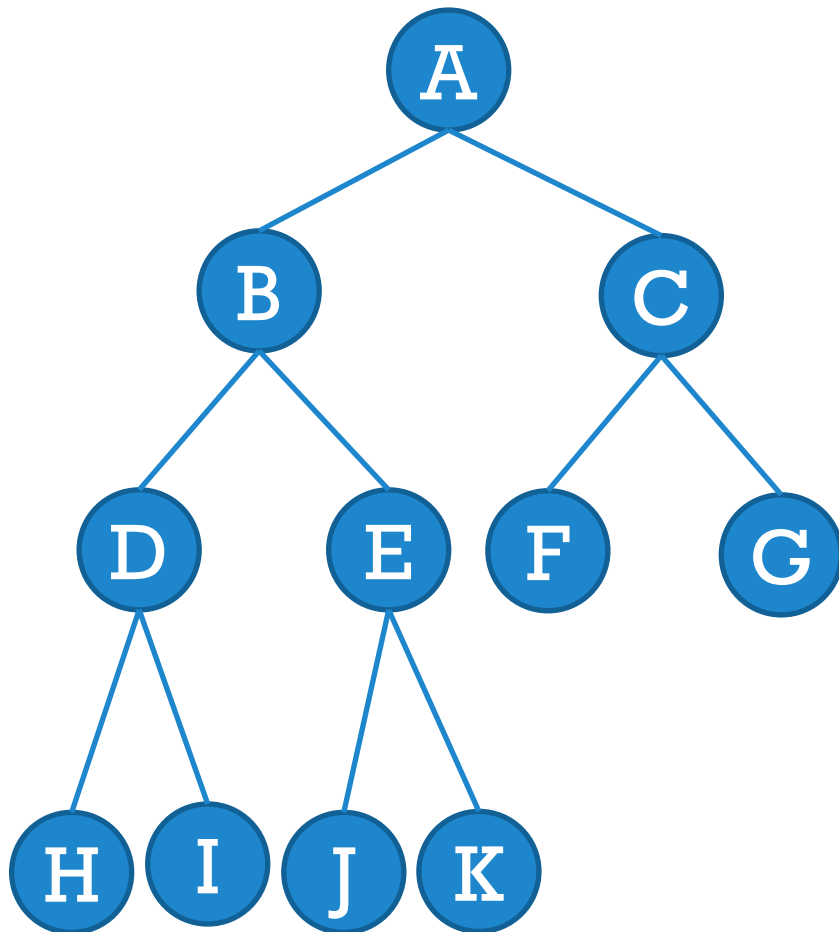
ÁRVORES BINÁRIAS QUASE COMPLETA



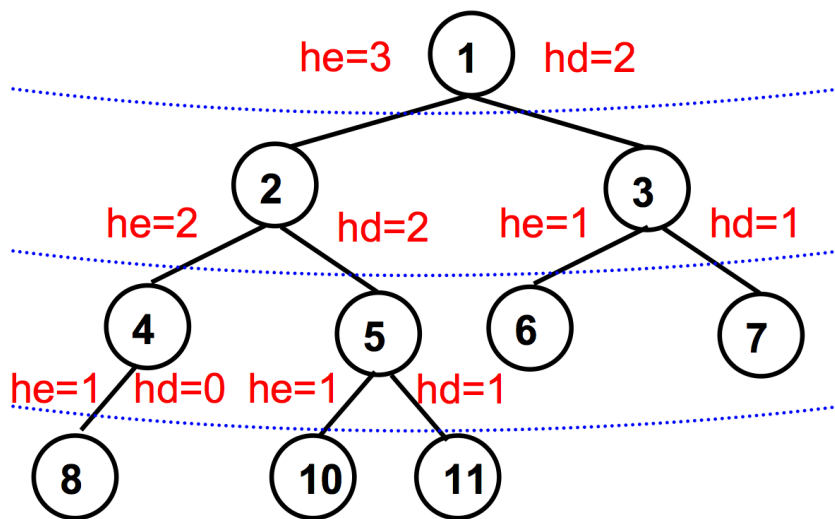
+

ÁRVORES BINÁRIAS QUASE COMPLETA

45



- Uma **árvore balanceada** é aquela onde para cada nó, as alturas de suas duas sub-árvores diferem de, no máximo, 1.



Árvore Binária Balanceada

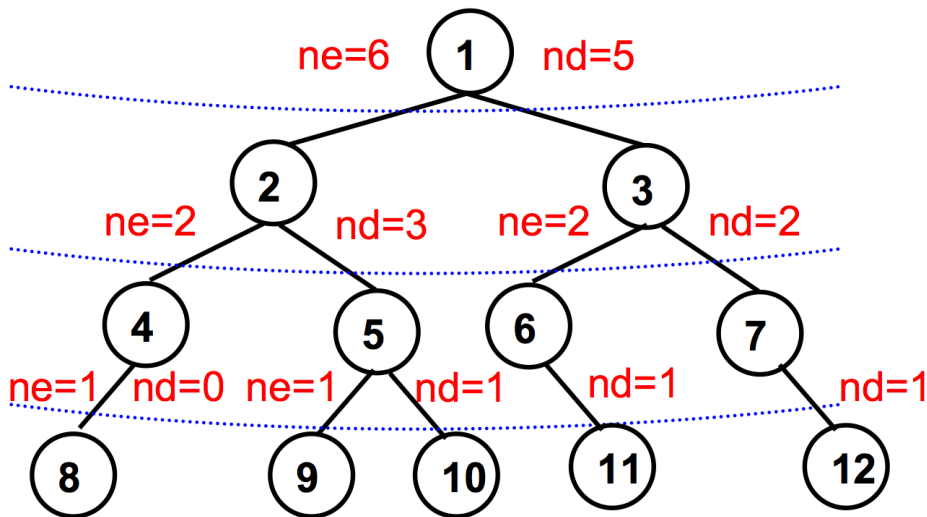
Considere

$he(x)$ como altura da sub-árvore esquerda e
 $hd(x)$ como a altura da sub-árvore direita

Para cada nível a diferença entre as alturas das sub-árvores (**$abs(he-hd)$**) não ultrapassa 1.

+ ÁRVORE BINÁRIA PERFEITAMENTE BALANCEADA

- O número de nós de suas sub-árvores esquerda e direita difere em, no máximo, 1.




Árvore Binária Perfeitamente Balanceada

Considere

$ne(x)$ como o número de nós da sub-
árvore esquerda, e

$nd(x)$ como o número de nós da sub-
árvore direita.

Para cada nível a diferença entre as
quantidades de nós das sub-árvores
(**$abs(ne-nd)$**) não ultrapassa 1.



+ ÁRVORES PERCURSOS

- Uma vez que determinadas informações são armazenadas é necessário alguma maneira acessar as informações lá contidas.
- Em uma lista linear, bastaria percorrer essa lista do início ao fim, acessando um elemento após o outro de forma seqüencial.



PERCURSOS

50

- Entretanto, uma árvore é uma estrutura não linear e dessa maneira temos algumas considerações:
 - por onde iniciar o percurso?
 - onde encerrar?

- Percorrer uma árvore binária ‘visitando’ cada nó uma única vez.
- Um percurso gera uma seqüência linear de nós, e dessa forma é possível falar de nó predecessor ou sucessor de um nó, segundo um dado percurso.



PERCURSOS

- Não existe um percurso único para árvores;
- Diferentes percursos podem ser realizados, dependendo da aplicação.
- A utilização é imprimir uma árvore, remover um item, buscar por um item, entre outras.

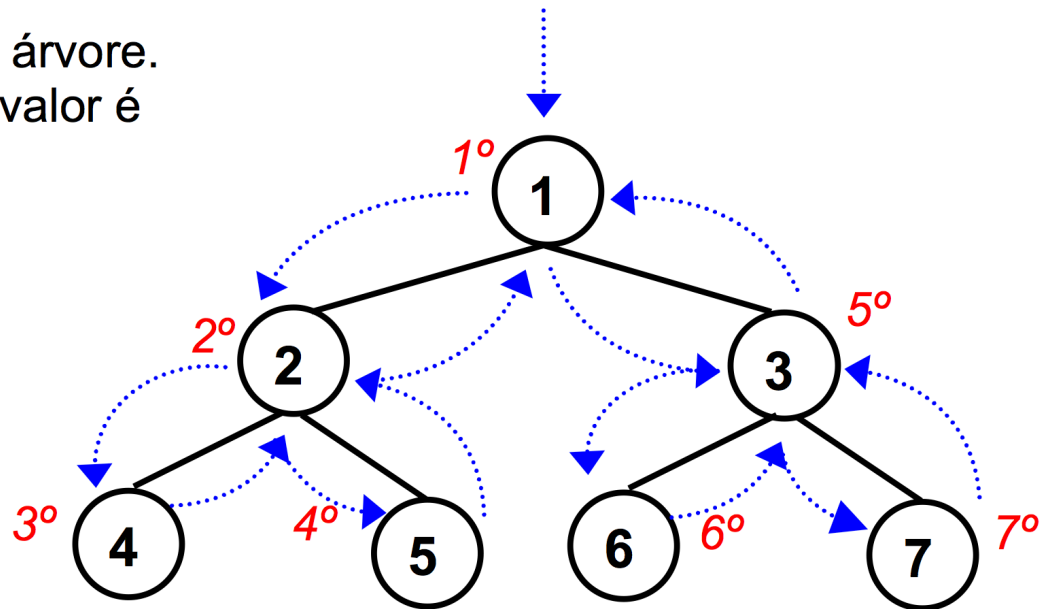
- Há **três** percursos básicos para árvores binárias:
 - pré-ordem (*Pre-order*) ou profundidade;
 - em-ordem (*In-order*) ou ordem simétrica;
 - pós-ordem (*Post-order*).
- A diferença entre esses percursos é, basicamente, a ordem em que os nós são o “**visitados**”.

+ PERCURSO PRÉ-ORDEM ou PROFUNDIDADE

■ O percurso pré-ordem consiste nos seguintes passos:

- Mostra o valor do nó;
- Visita o nó esquerdo;
- Visita o nó direito;

Inicia o percurso pela raiz da árvore. Assim que o nó é visitado, o valor é mostrado (1ª passagem).



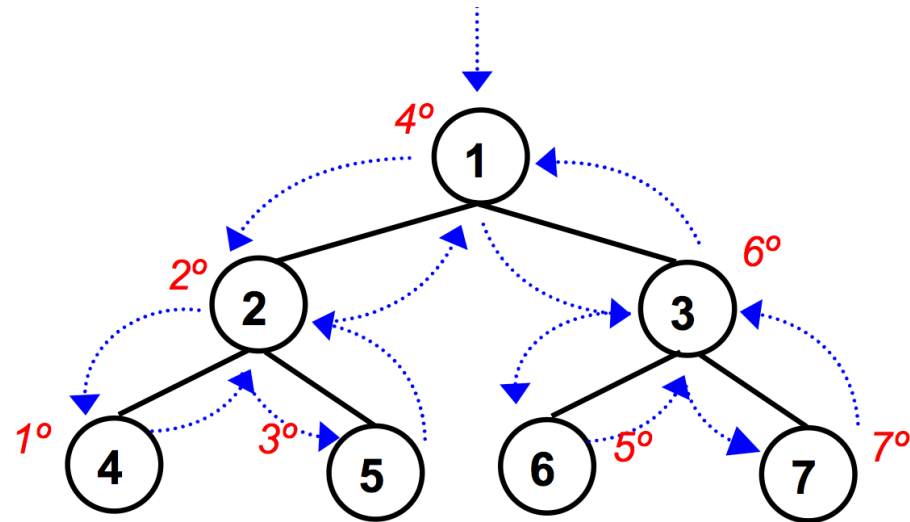
Resultado do Percurso: 1,2,4,5,3,6,7

+ PERCURSO EM-ORDEM ou ORDEM SIMÉTRICA

■ O percurso em-ordem consiste nos seguintes passos:

- Visita o nó esquerdo;
- Mostra o valor do nó;
- Visita o nó direito;

Inicia o percurso pela raiz da árvore.
Caminha inicialmente pelos nós da esquerda, só exibindo os valores quando todos à esquerda já tiverem sido visitados (2ª passagem).



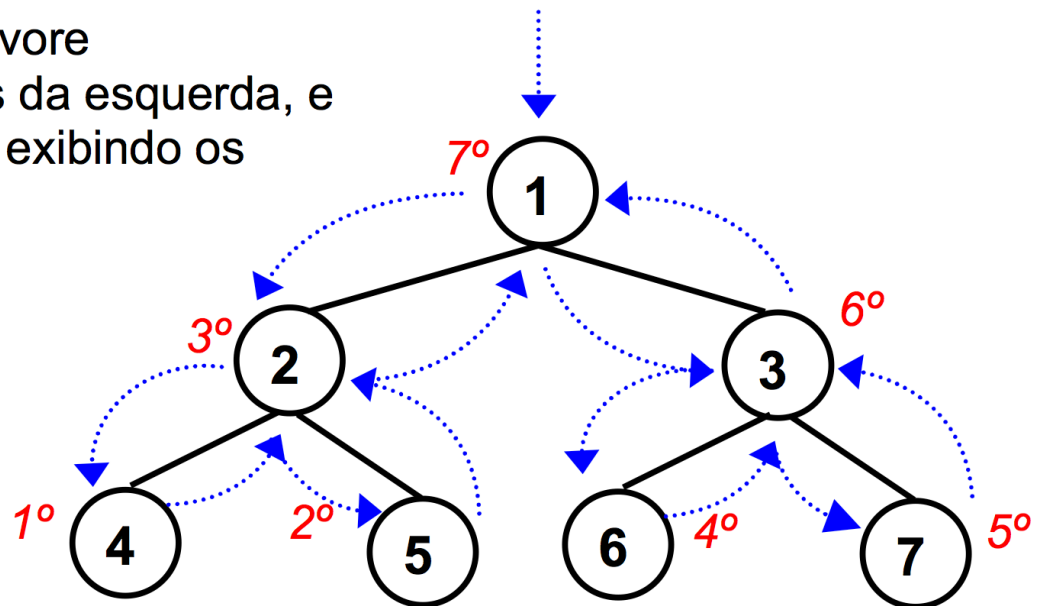
Resultado do Percurso: 4,2,5,1,6,3,7

■ O percurso pós-ordem consiste nos seguintes passos:

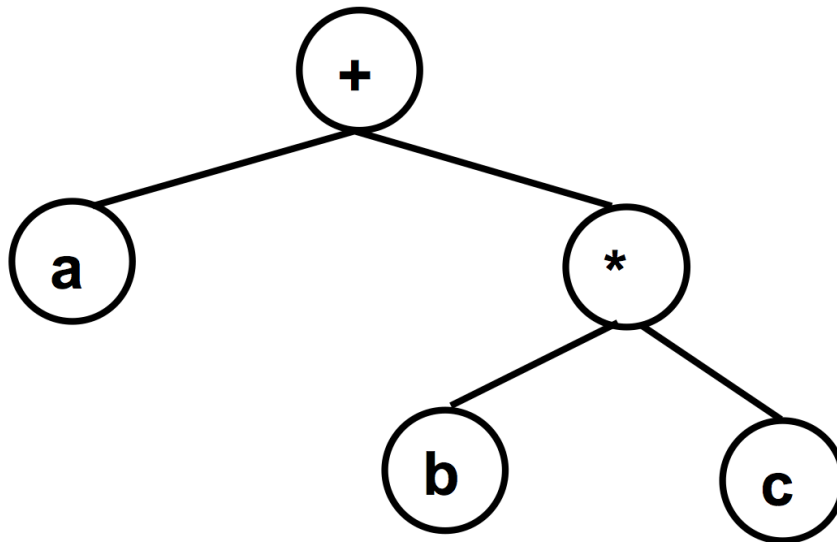
- Visita o nó esquerdo;
- Visita o nó direito;
- Mostra o valor do nó;

Inicia o percurso pela raiz da árvore

Caminha inicialmente pelos nós da esquerda, e em seguida pelos da direita, só exibindo os valores quando todos os nós descendentes já tiverem sido visitados (3ª passagem).



Resultado do Percurso: 4,5,2,6,7,3,1



Pré-ordem = +a*bc
Em-ordem = a+(b*c)
Pós-ordem = abc*+

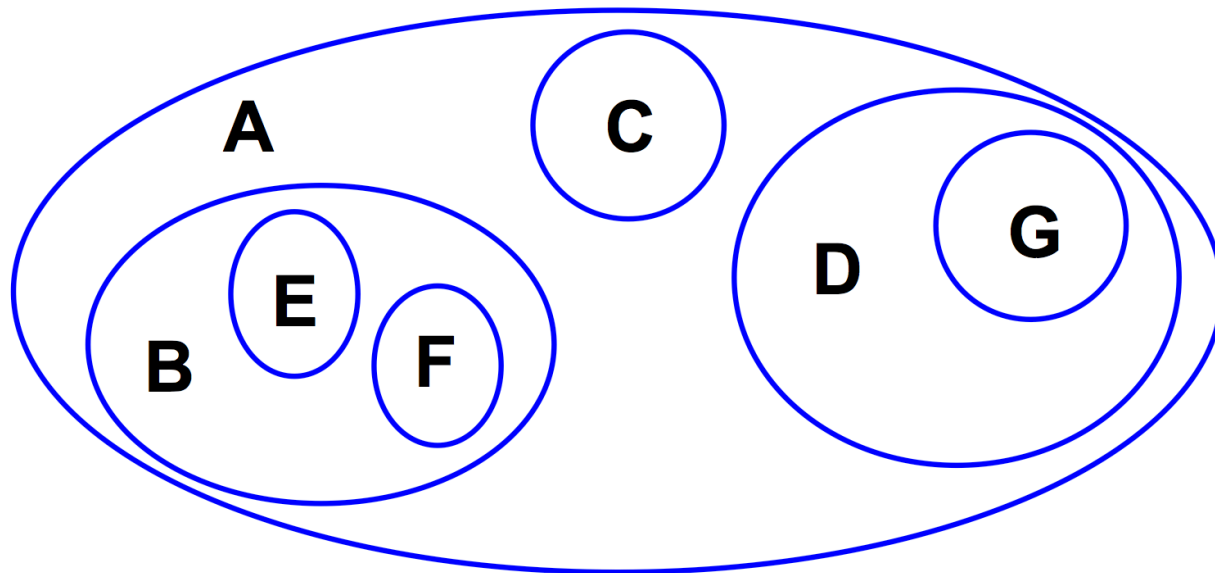
- Esses percursos são implementados recursivamente, pois permite um algoritmo simples, versátil e que não compromete o desempenho da máquina, pois em geral tais estruturas, quando balanceadas, se utilizam de uma pilha de recursão correspondente à altura da árvore.
- Em algoritmos iterativos é preciso o uso de uma pilha, ou então uma referência em cada nó ao nó Pai respectivo.



+ LISTA DE EXERCÍCIOS

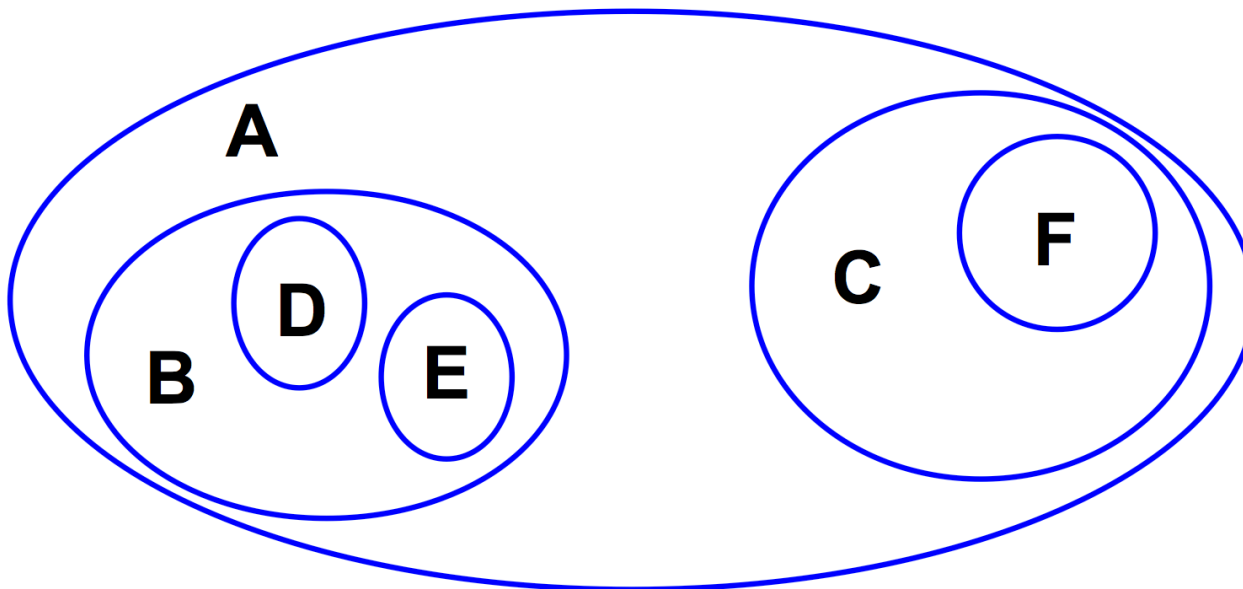
- Dado o seguinte Diagrama de Venn, construa a respectiva representação em forma de parênteses e árvore.

a)



- Dado o seguinte Diagrama de Venn, construa a respectiva representação em forma de parênteses e árvore.

b)





#EXERCÍCIO 2

Para cada árvore gerada no #exercício 1 responda:

- 1) Descreva para cada árvore:
 - a) Grau dos nós;
 - b) Grau da árvore;
 - c) Folhas da árvore;
 - d) Raiz da árvore;
 - e) Nós em cada nível;
 - f) Altura da árvore;

- 2) É uma árvore binária? Se for binária:
 - a) ela é estritamente binária?
 - b) ela é completa ou quase completa?
 - c) ela está balanceada? Ela é perfeitamente balanceada?



#EXERCÍCIO 3

Para cada árvore gerada no #exercício 1 faça:

- a) pré-ordem (*Pre-order*) ou profundidade;
- b) em-ordem (*In-order*) ou ordem simétrica;
- c) pós-ordem (*Post-order*).



That's all Folks!