

**INSTITUTO FEDERAL**  
Sul-rio-grandense

Câmpus  
Pelotas

EDUCAÇÃO  
**PÚBLICA**  
**100%**  
GRATUITA

# Estrutura de Dados

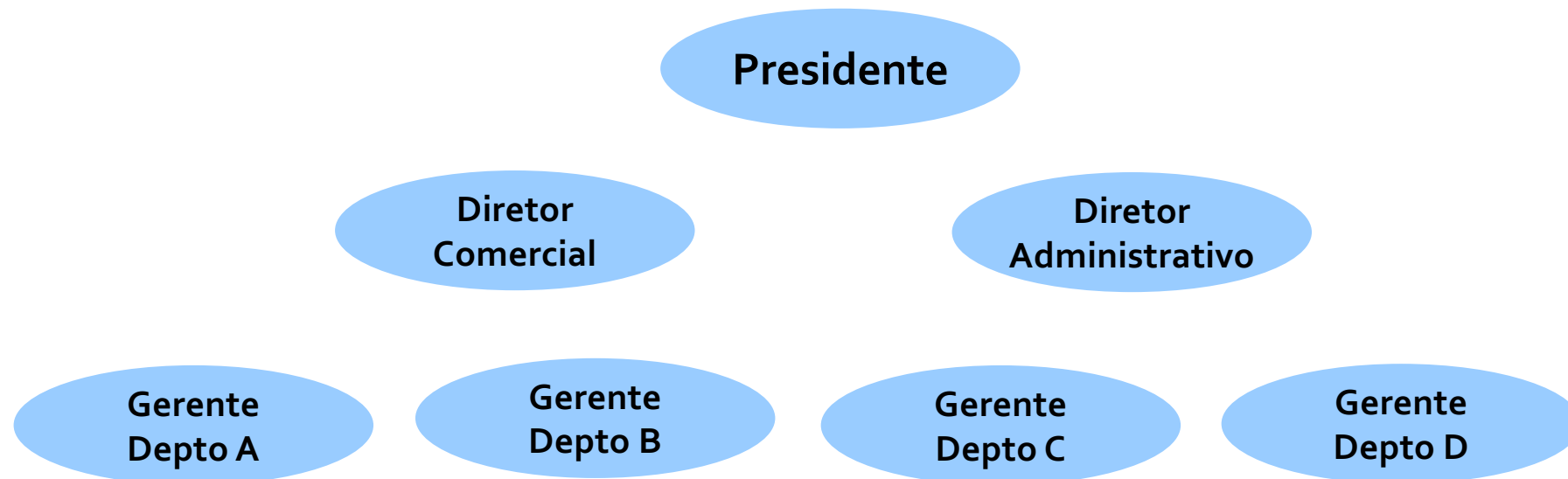
Aula 11

## Árvores

# Árvores

Uma Árvore é uma estrutura de dados que se caracteriza por uma relação de hierarquia entre os elementos que a compõe.

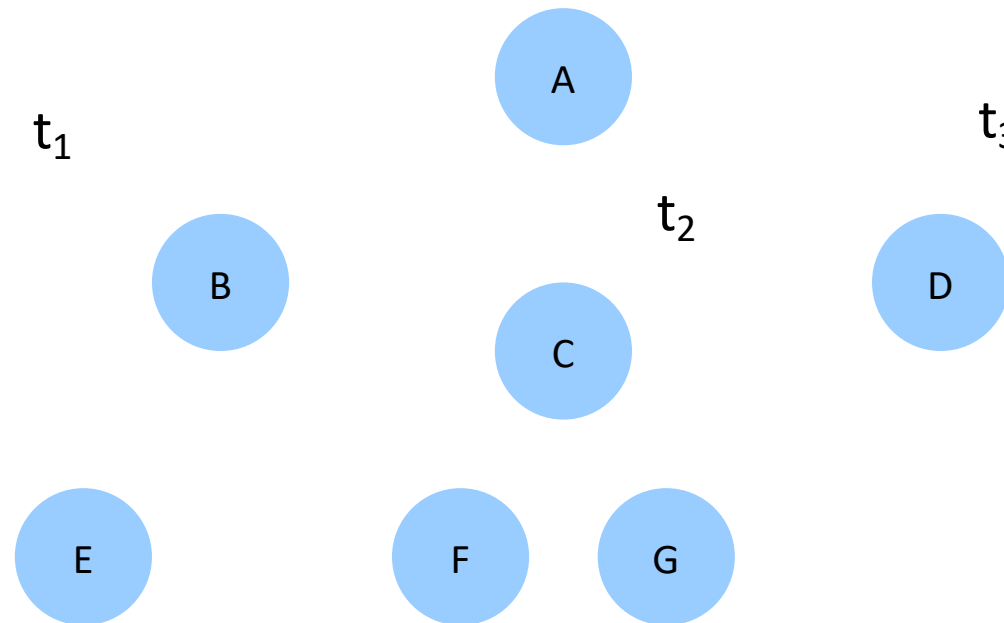
- Ex:
- Organograma de uma empresa.
  - Divisão de um livro em capítulos, seções, tópicos.
  - Estruturas de diretórios.



# Árvores

**Definição:** É um conjunto finito  $T$  de um ou mais nodos tal que:

- Existe um nodo denominado raiz da árvore.
- Os demais nodos formam  $n \geq 0$  subconjuntos disjuntos  $t_1, t_2, \dots, t_n$ , onde cada um deles é uma árvore. As árvores  $t_i$ , ( $1 \leq i \leq n$ ) recebem a denominação de subárvores.



# Árvores

## Terminologia:

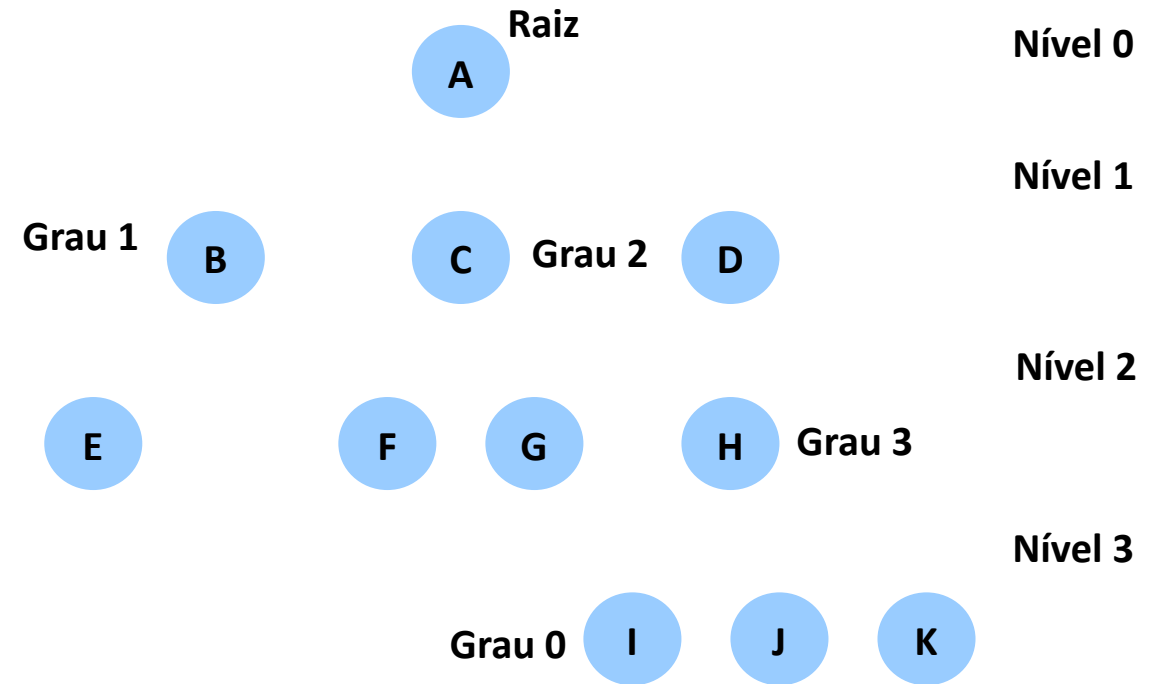
**Raiz:** Nodo de origem da árvore.

**Folha (nodo Terminal):** Nodo que não tem filhos.

**Grau de um nodo:** número de filhos do nodo.

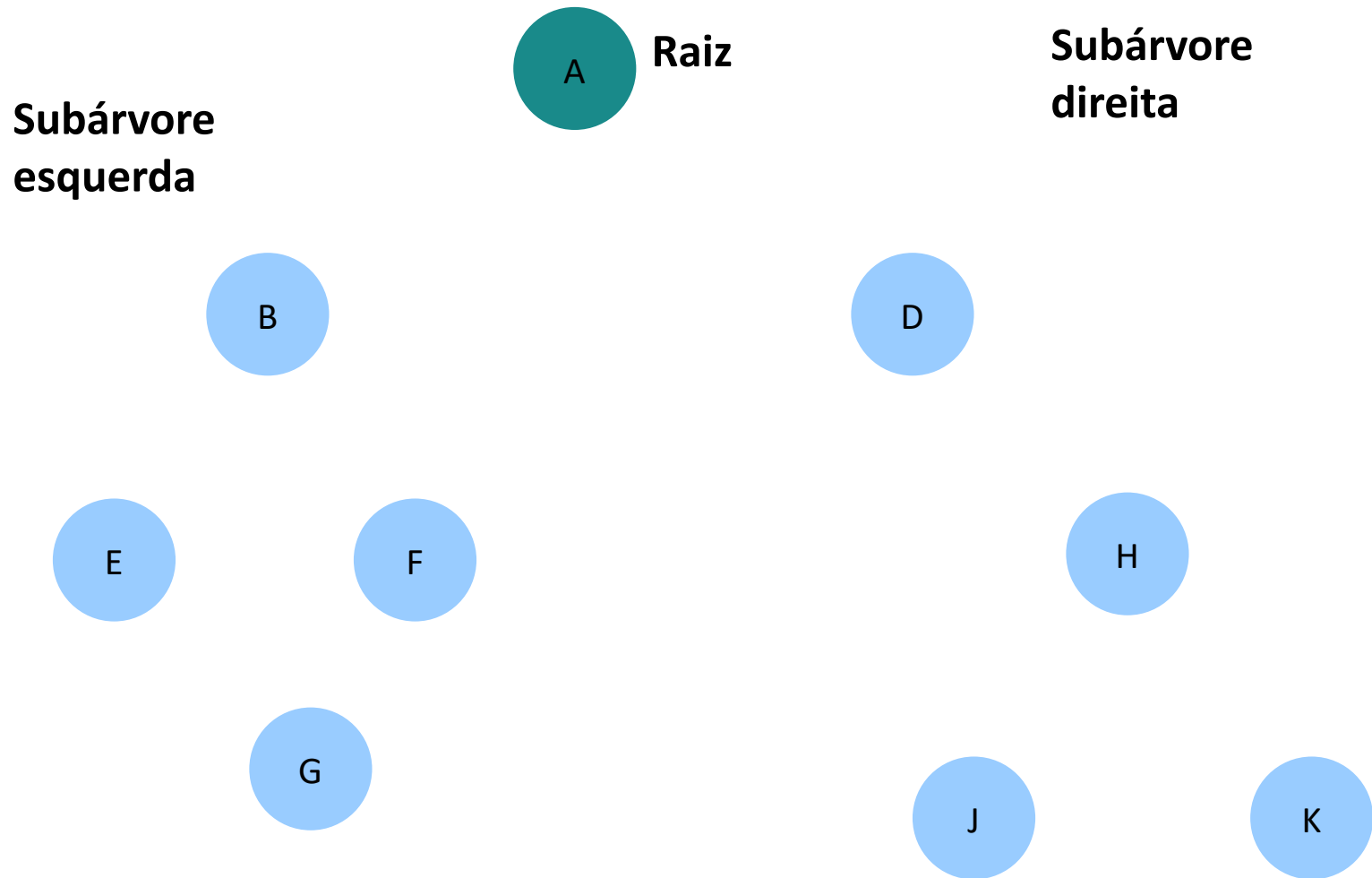
**Nível de um nodo:** zero para o nodo raiz; e para os demais nodos é o número de “linhas” que o ligam ao nodo raiz.

**Altura (ou Profundidade):** é o nível mais alto da árvore.



# Árvores Binárias

São árvores em que todos os nodos tem **grau menor ou igual a dois**.



# Percursos em Árvores Binárias

Define em que ordem os nodos da Árvore Binária são visitados.

1. Pré Ordem
2. Em Ordem
3. Pós Ordem
4. Em Nível



# 1. Percurso em Pré Ordem

inicio

se árvore não é vazia então

Visitar o nodo raiz

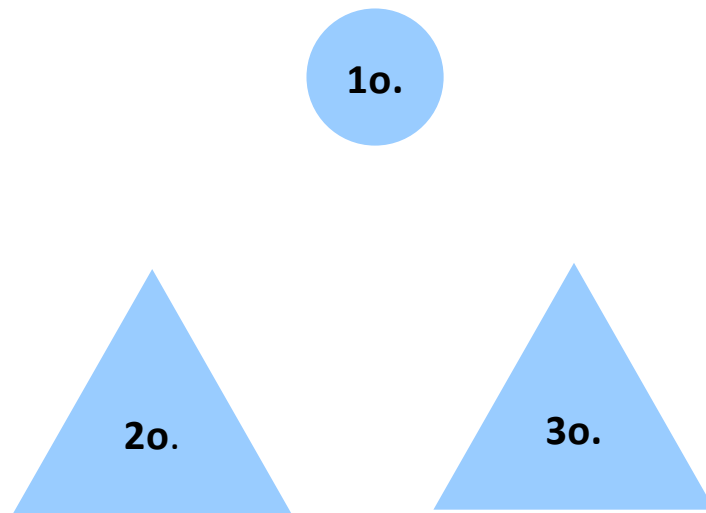
Percorrer a subárvore esquerda

Percorrer a subárvore direita

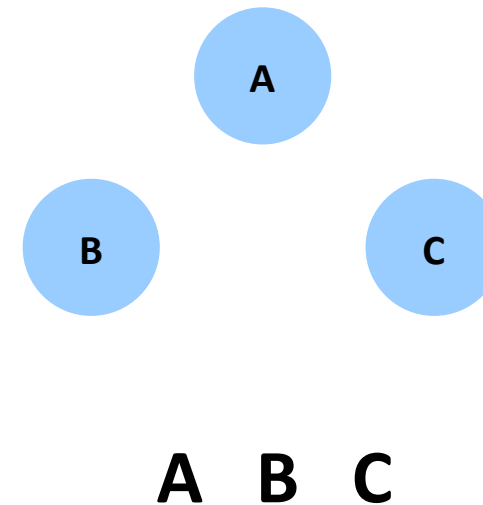
fim se

fim

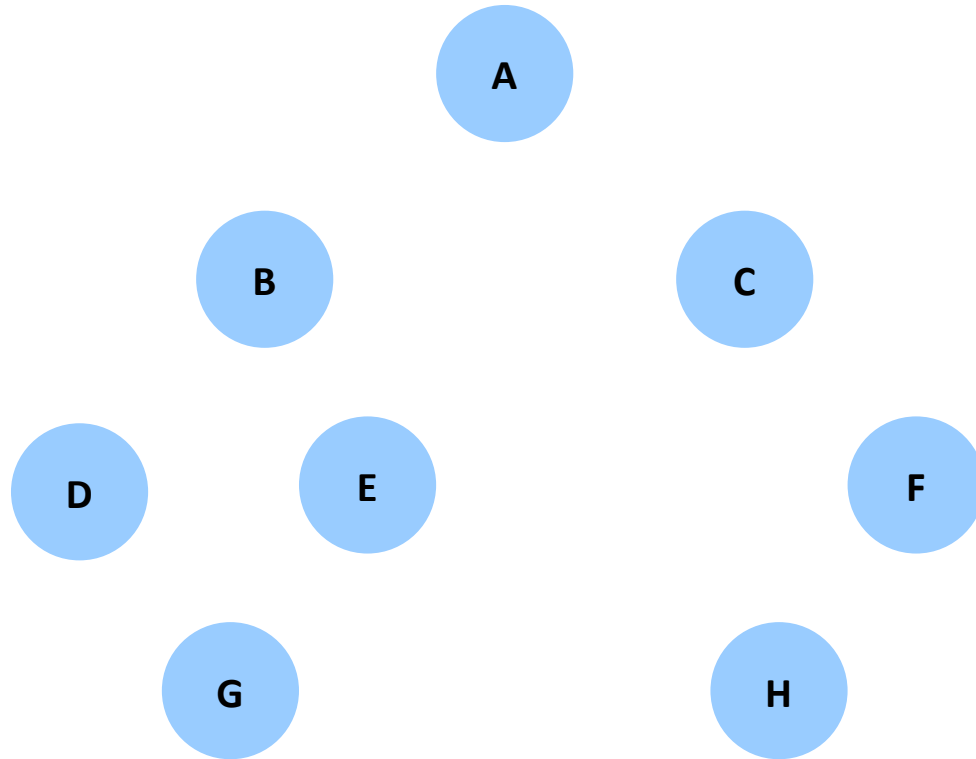
**R – E – D**



Exemplo



# 1. Percurso em Pré Ordem



**R – E – D**

Percurso:

A B D E G C F H

? ? ?



## 2. Percurso Em Ordem

inicio

se árvore não é vazia então

Percorrer a subárvore esquerda

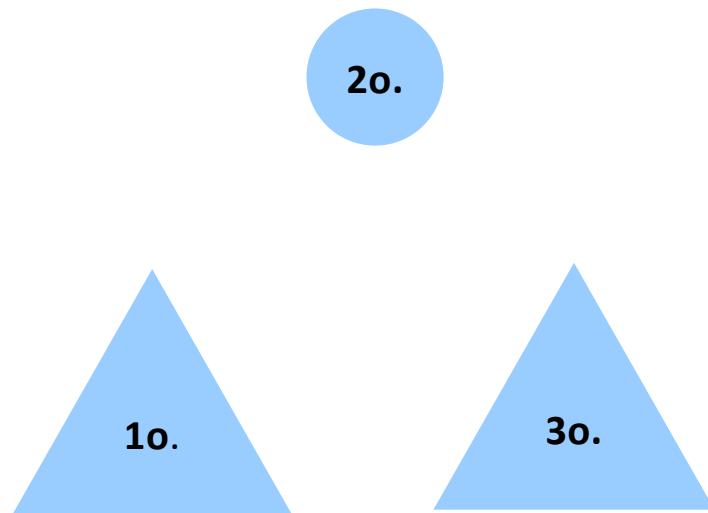
Visitar o nodo raiz

Percorrer a subárvore direita

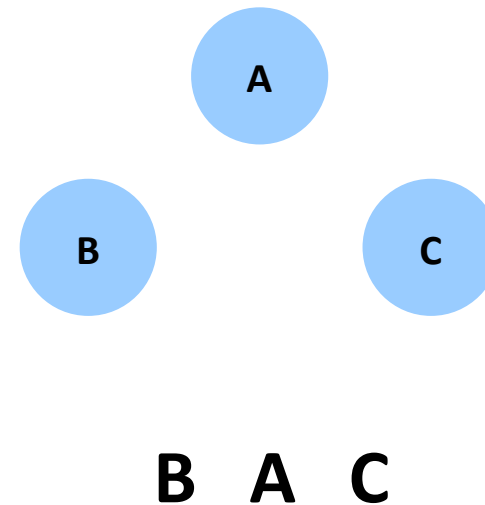
fim se

fim

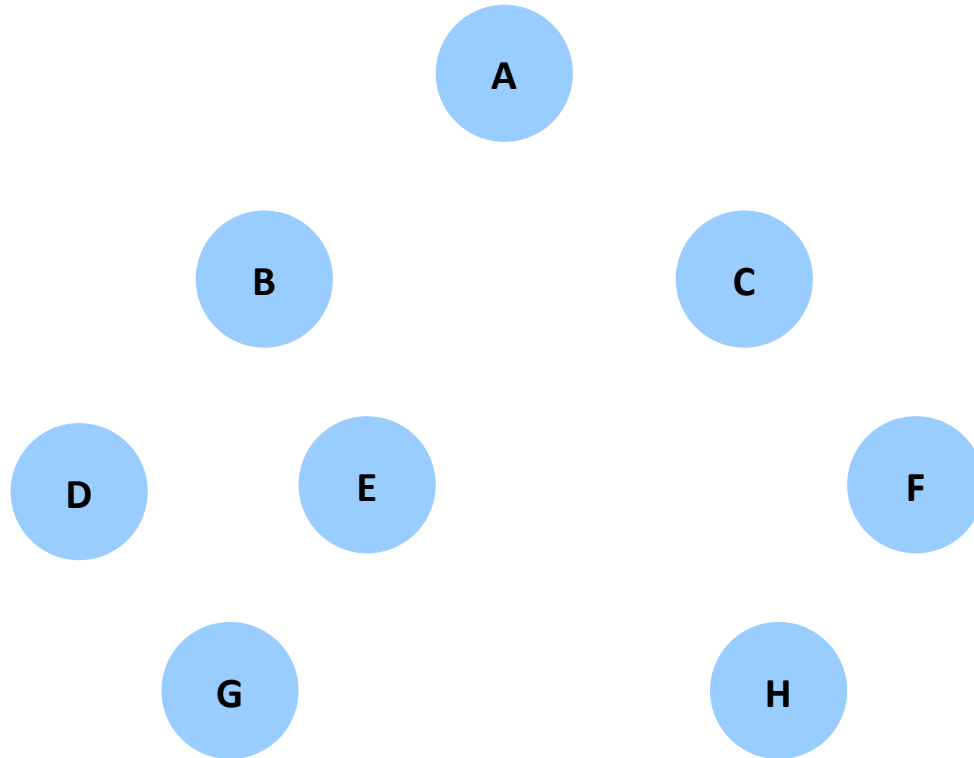
**E – R – D**



Exemplo



## 2. Percurso Em Ordem



? ? ?

**E – R – D**

Percurso:

D B G E A C H F



### 3. Percurso em Pós Ordem

inicio

se árvore não é vazia então

Percorrer a subárvore esquerda

Percorrer a subárvore direita

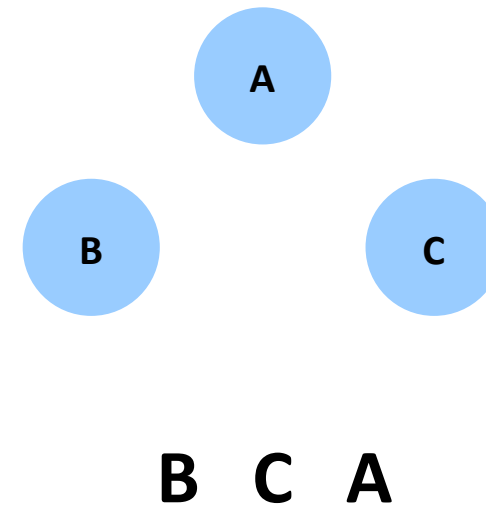
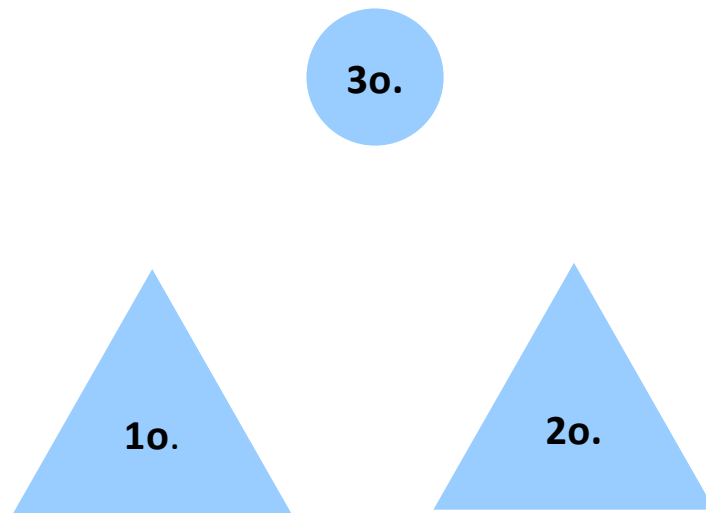
Visitar o nodo raiz

fim se

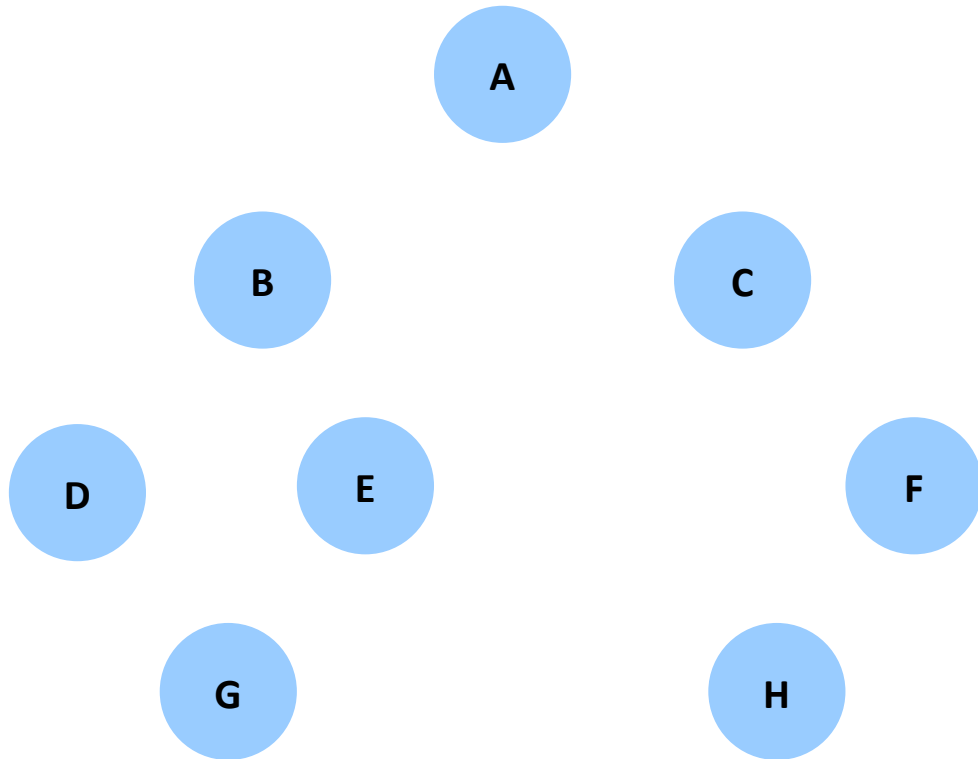
fim

**E – D – R**

Exemplo



### 3. Percurso em Pós Ordem



? ? ?

**E – D – R**

Percurso:

D G E B H F C A



EDUCAÇÃO  
PÚBLICA  
**100%**  
GRATUITA

## 4. Percurso em Nível

inicio

Inserir o nodo raiz em uma fila

Enquanto a fila não estiver vazia

Retirar o nodo T da fila

Visitar T

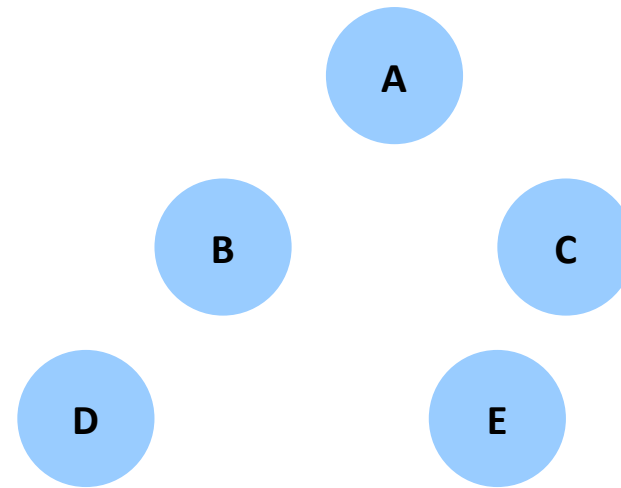
Adicionar os filhos de T na fila

fim Enquanto

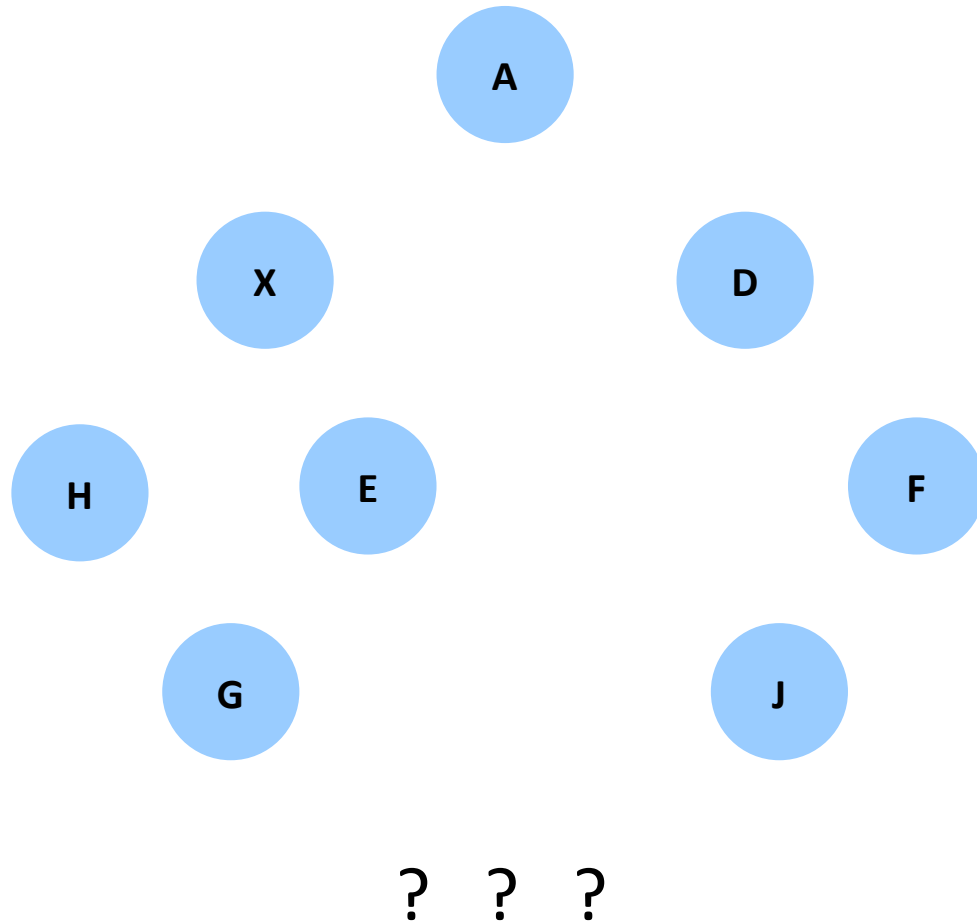
fim

Exemplo

A B C D E



## 4. Percurso em Nível



inicio

Inserir o nodo raiz em uma fila

Enquanto a fila não estiver vazia

Retirar o nodo T da fila

Visitar T

Adicionar os filhos de T na fila

fim Enquanto

fim

Percurso:

A X D H E F G J



# Árvores - Implementação

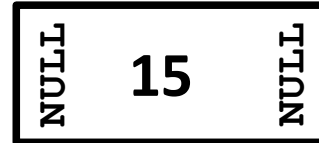
Representação por encadeamento:

raiz  
1200

esq 1200 dir



esq 1300 dir



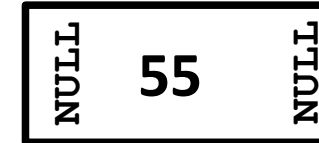
esq 1000 dir



esq 1100 dir

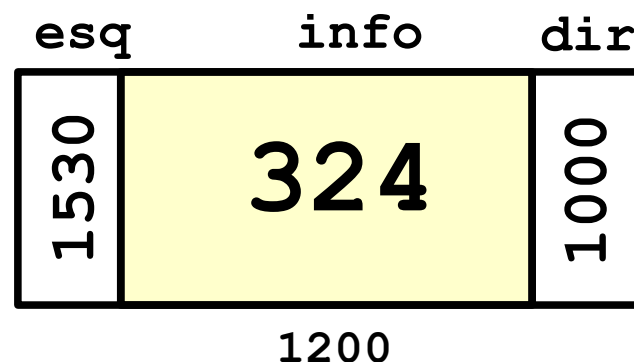


esq 2200 dir



# Árvores - Implementação

Nodo para armazenar um inteiro



**esq:** contém o endereço do nodo filho a esquerda (**NULL** se não possui filho a esquerda).

**info :** contém a informação armazenada (Ex: um inteiro).

**dir:** contém o endereço do nodo filho a direita (**NULL** se não possui filho a direita).



# Árvores - Implementação

```
typedef struct {  
    int cod;  
    float sal;  
} Dado;
```

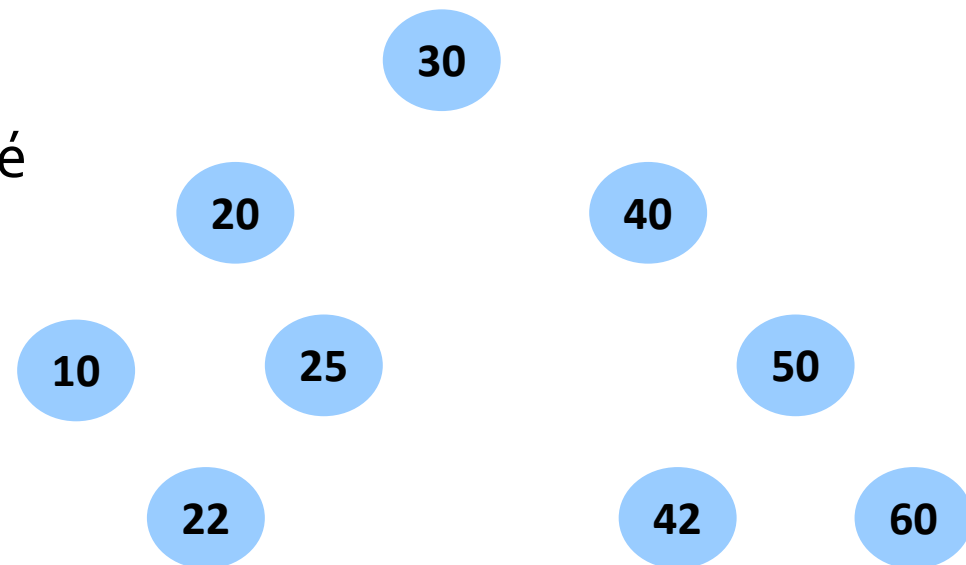
```
typedef struct nodo Nodo;
```

```
struct nodo {  
    Dado info;  
    Nodo *esq;  
    Nodo *dir;  
};
```

```
typedef struct {  
    Nodo *raiz;  
} Arvore;
```

# Árvore de Busca Binária

- Os dados são distribuídos pelos nodos de forma a facilitar a pesquisa de um determinado elemento.
- Uma árvore binária, cuja raiz armazena o elemento R, é denominada árvore de busca binária se:
  - 1) **Todo** elemento armazenado na **subárvore esquerda é menor** que R.
  - 2) **Nenhum** elemento armazenado na **subárvore direita é menor** que R.
  - 3) As árvores esquerda e direita também são árvores de busca binária.



# Árvore de Busca Binária - Inserção

início

se a árvore está vazia

então

inserir o nodo

senão

se o dado do nodo que será inserido é **menor**  
que o dado armazenado no nodo raiz

então

inserir o nodo na subárvore **esquerda**

senão

inserir o nodo na subárvore **direita**

fim\_se

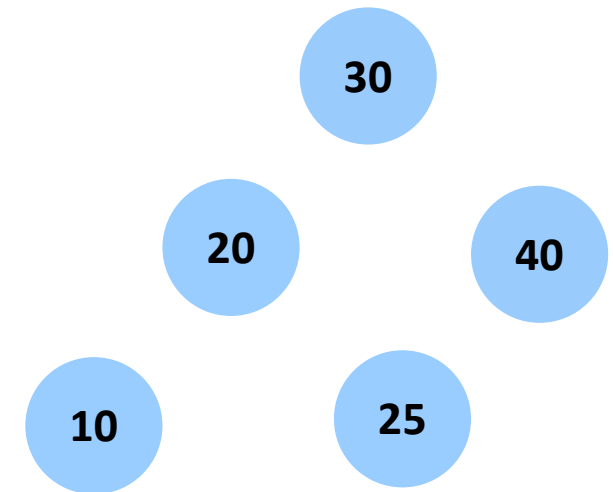
fim\_se

fim

Inserir os nodos:

**30 20 25 10 40**

em uma árvore vazia:



EDUCAÇÃO  
PÚBLICA  
**100%**  
GRATUITA

# Árvore de Busca Binária - Procura

início

se a árvore está vazia então

nodo não encontrado

senão

se a raiz armazena o elemento procurado então

nodo encontrado

senão

se o valor procurado for menor que o valor  
armazenado no nodo raiz então

procurar a partir da subárvore esquerda

senão

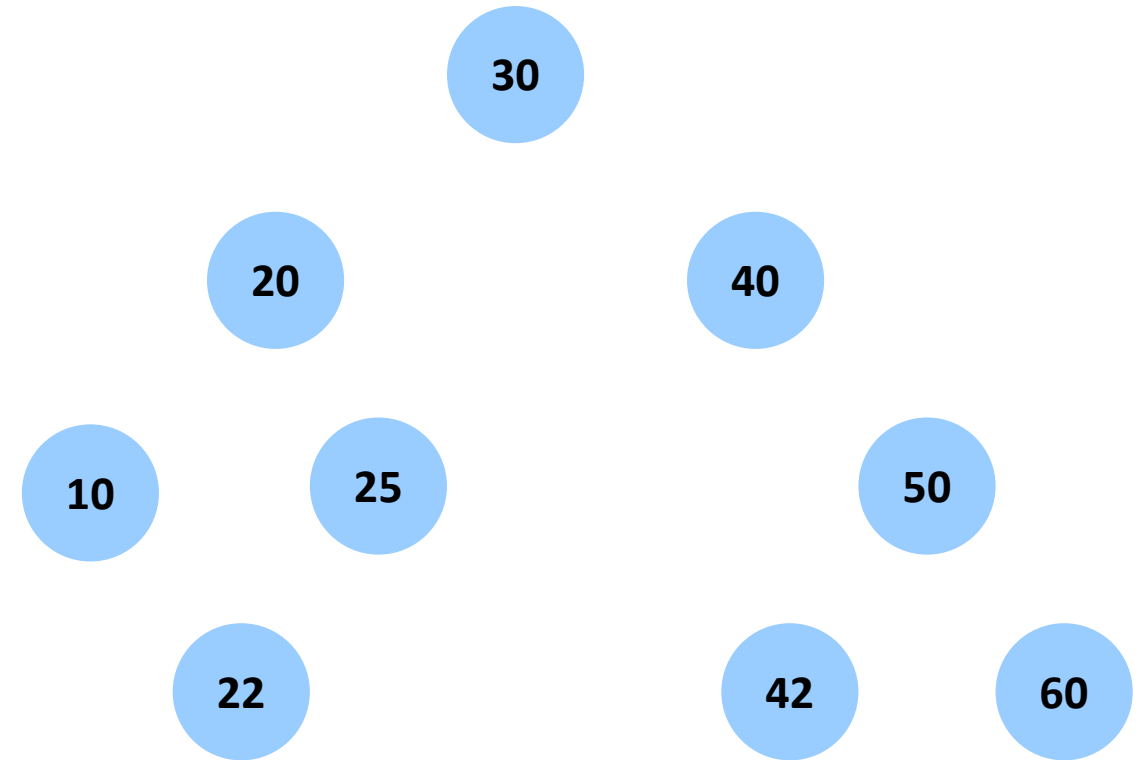
procurar a partir da subárvore direita

fim\_se

fim\_se

fim\_se

fim



Procurar os nodos:

**27**

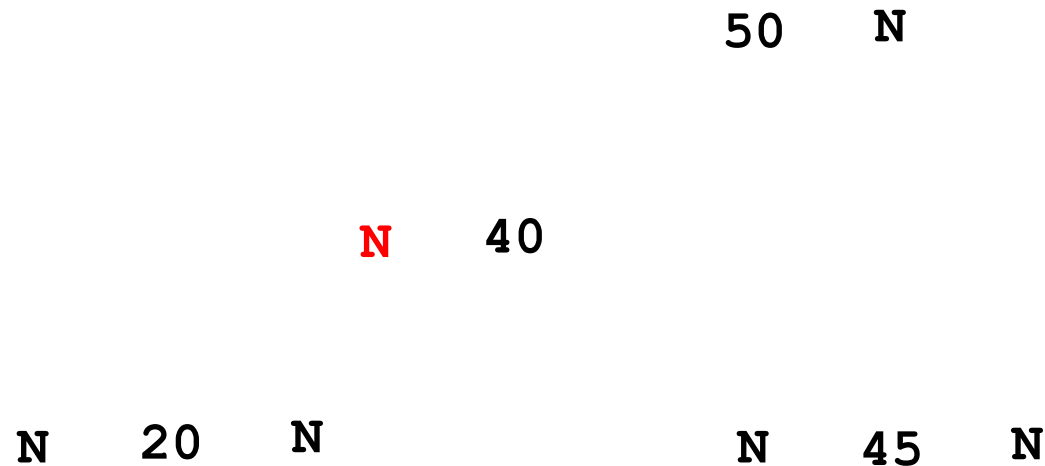
**25**

# Árvore de Busca Binária - Remoção

Procurar o elemento a ser removido. Caso seja encontrado:

Existem 3 casos:

1) O elemento removido não possui filhos. Remover o nodo e tornar nula a raiz da subárvore.



# Árvore de Busca Binária - Remoção

2) O elemento removido possui um filho. Remover o nodo substituindo-o pelo seu nodo filho.

N 50

70

N 60

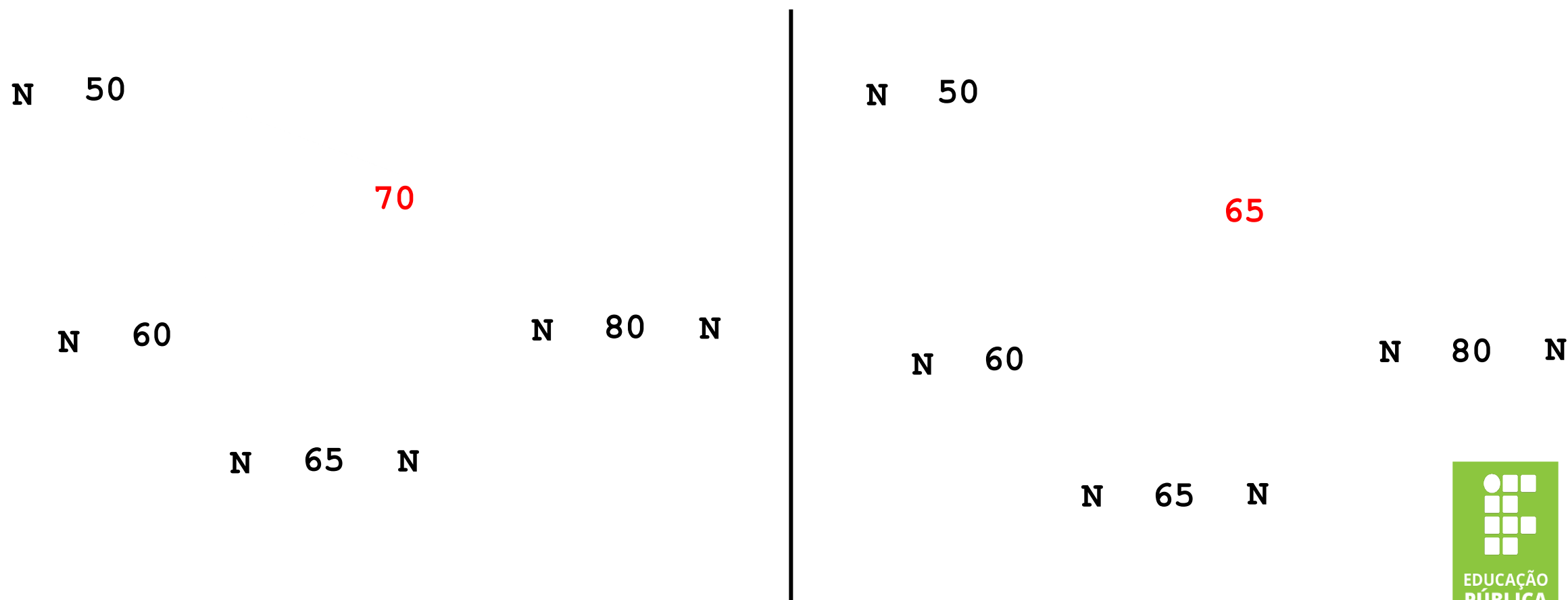
N 80 N

N 65 N



# Árvore de Busca Binária - Remoção

3) O elemento removido possui dois filhos. Procurar o nodo que armazena o maior elemento na subárvore esquerda. Este nodo será removido e o elemento armazenado será copiado para o nodo excluído.

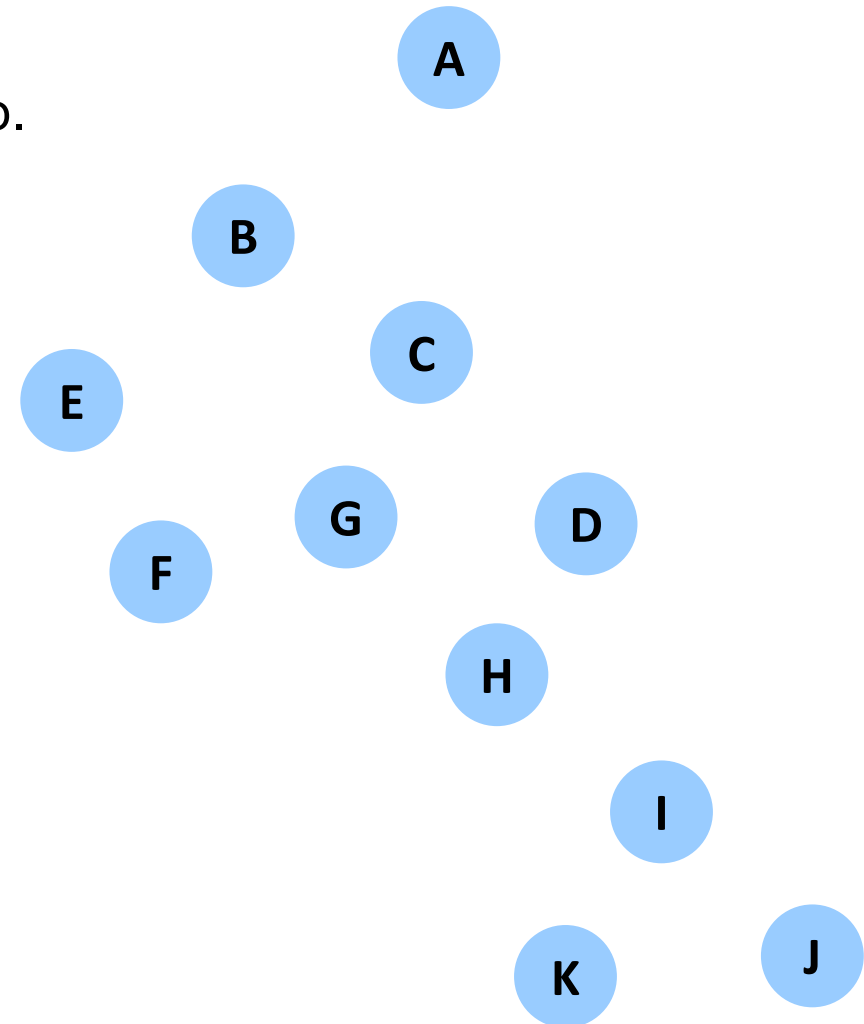
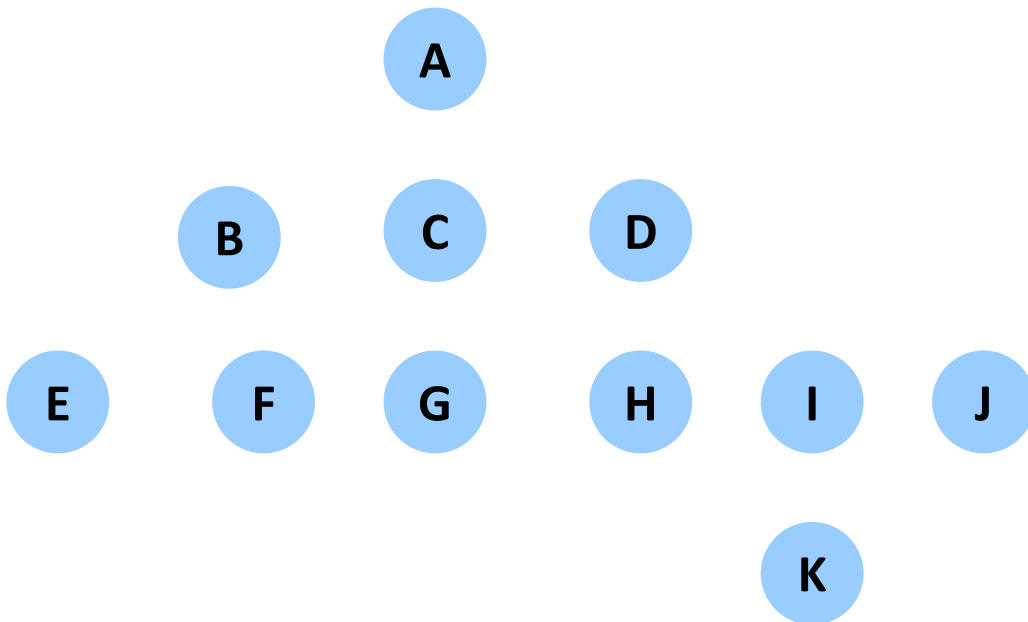


# Árvores de Grau Qualquer

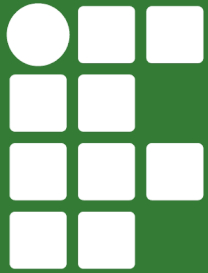
Para representar árvores de grau qualquer utilizar uma árvore binária fornecendo aos ponteiros **esq** e **dir** significados diferentes:

**esq** : passa a armazenar o endereço do primeiro filho.

**dir**: passa a armazenar o endereço do irmão.







**INSTITUTO FEDERAL**  
Sul-rio-grandense

Câmpus  
Pelotas

EDUCAÇÃO  
**PÚBLICA**  
**100%**  
GRATUITA

# Estrutura de Dados

Aula 11

## Árvores