

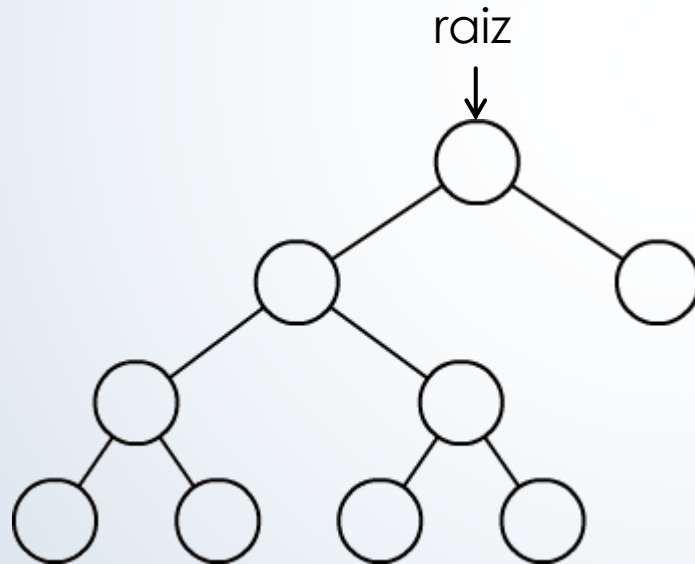
ESDC4: Estruturas de Dados I

Aula 07: Estruturas de Dados Não-Lineares - Árvores, Árvores Binárias e Árvores Binárias de Busca

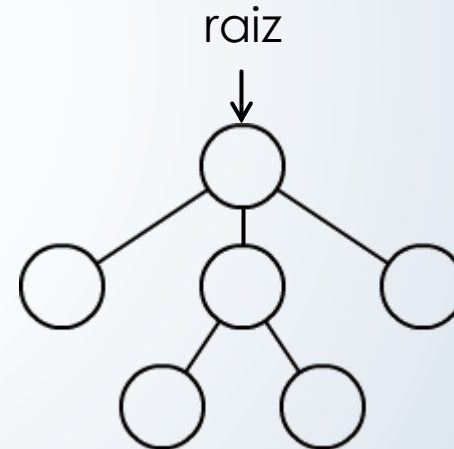
Bacharelado em Ciência da Computação
Prof. Dr. David Buzatto

- ▶ Árvores são grafos acíclicos com um nó ou vértice especial, denominado **raiz**.

Árvore 1



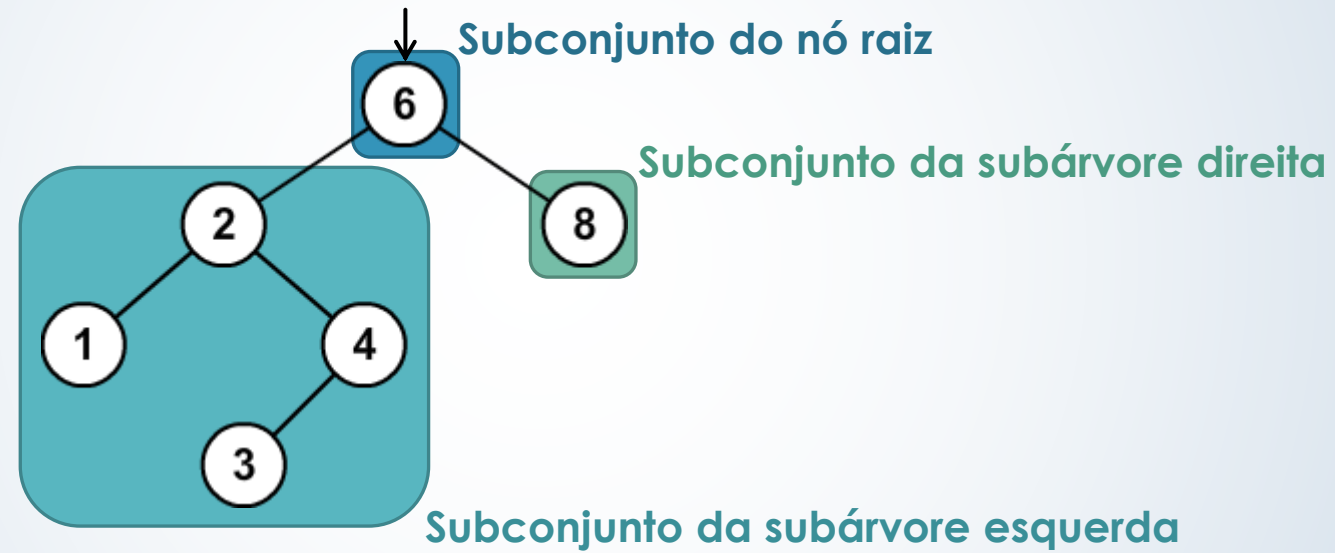
Árvore 2



Árvore Binária

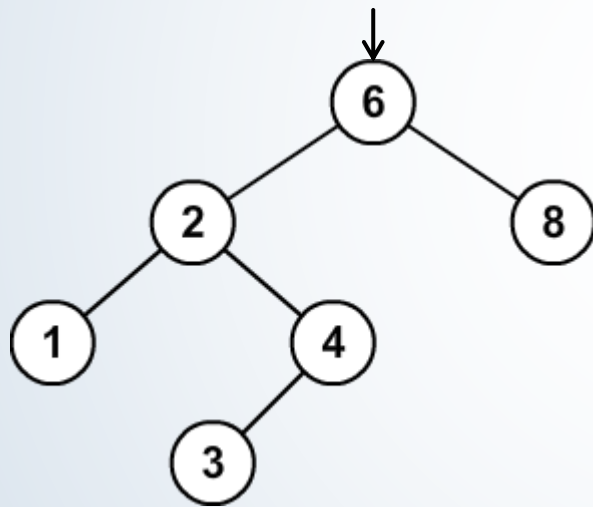
- Conjunto finito de elementos (nós);
- O primeiro nó é chamado de raiz;
- Formada por três subconjuntos:
 - Subconjunto do nó raiz;
 - Subconjunto da subárvore esquerda;
 - Subconjunto da subárvore direita;
- Implica em cada nó poder ter no máximo 2 filhos.

Árvore Binária

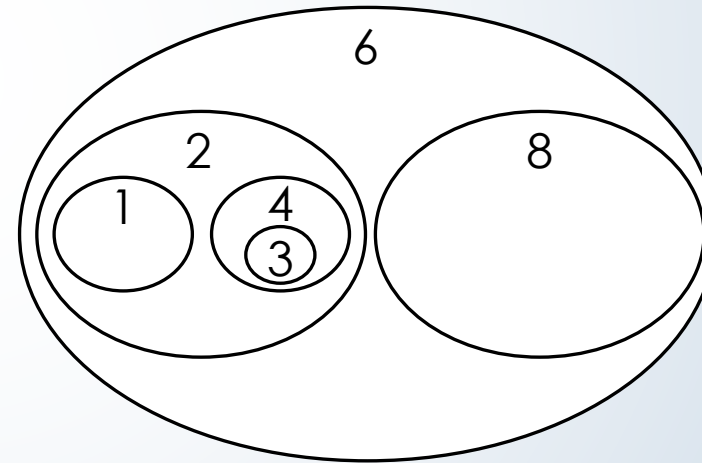


Árvore Binária

Representação em Árvore



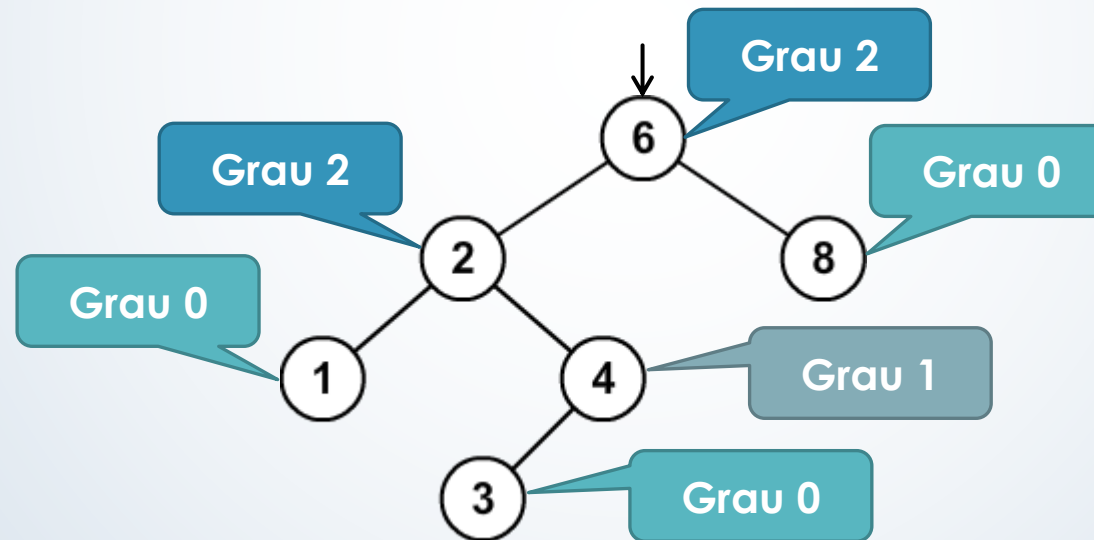
Representação em Conjunto



Árvore Binária

Propriedades

- ➔ O **grau** de um nó é igual ao seu número de subárvores;



Árvore Binária

Propriedades

- ▶ Em uma árvore binária, o grau máximo de um nó é 2;
- ▶ O grau de uma árvore é igual ao máximo dos graus de todos os seus nós;
- ▶ Uma árvore binária tem grau máximo igual a 2.

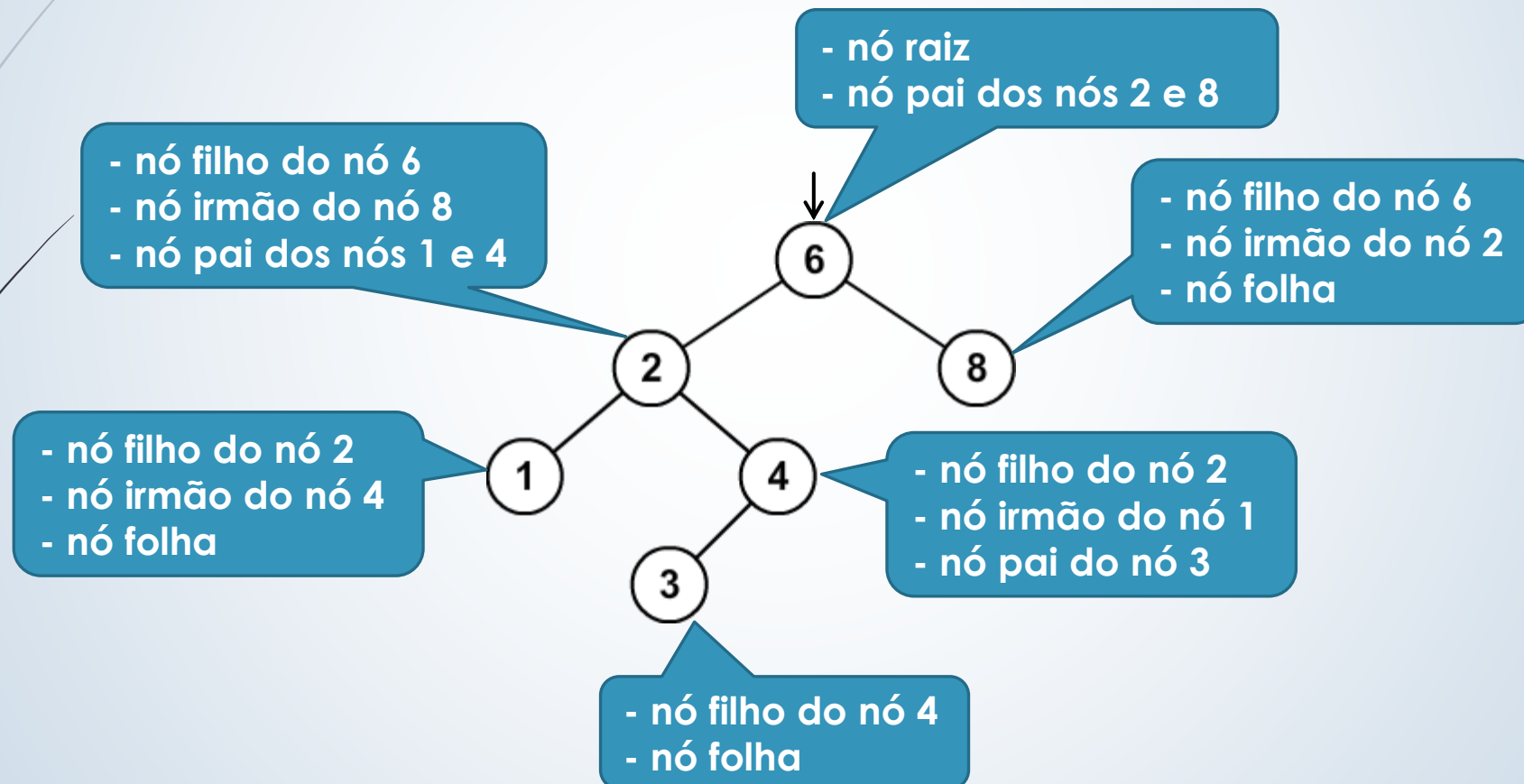
Árvore Binária

Propriedades

- **Nó pai:** nó acima e com ligação direta a outro nó;
- **Nó filho:** nó abaixo e com ligação direta a outro nó;
- **Nós irmãos:** nós que possuem o mesmo pai;
- **Nó folha ou terminal:** nó que não possui filhos;

Árvore Binária

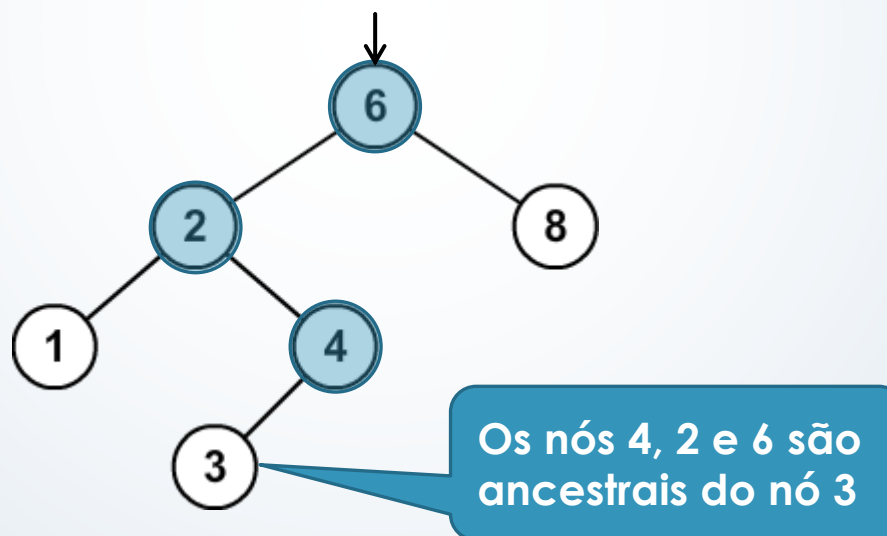
Propriedades



Árvore Binária

Propriedades

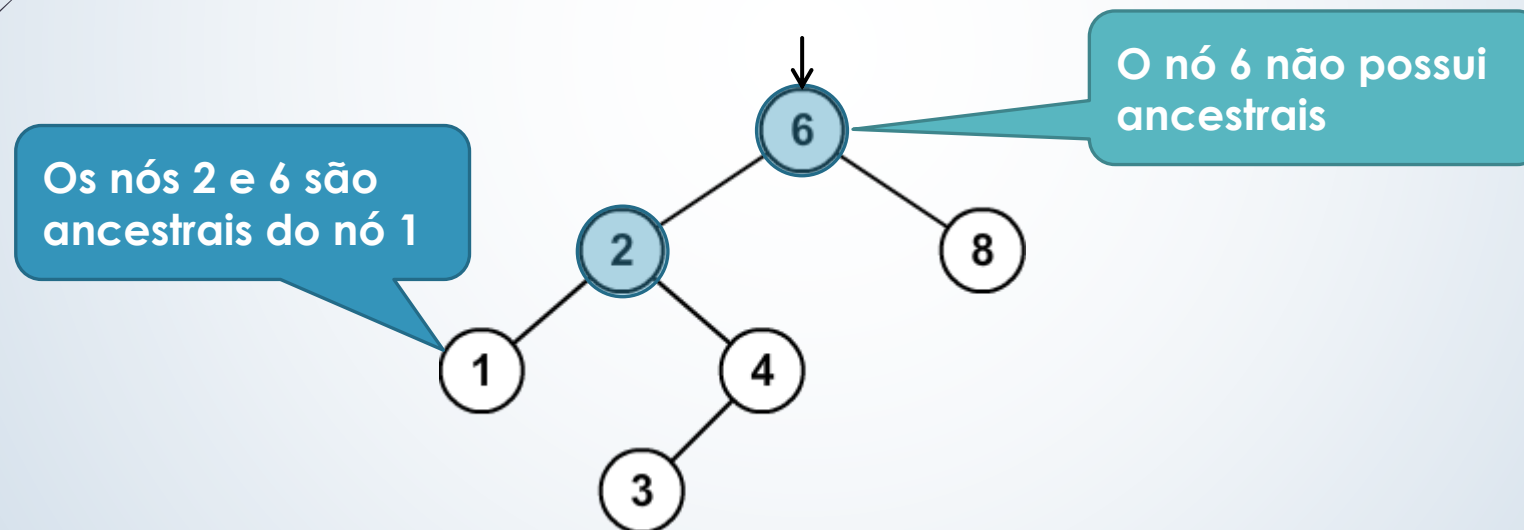
- **Nó ancestral:** nós que estão acima de um nó e possuem ligação direta ou indireta.



Árvore Binária

Propriedades

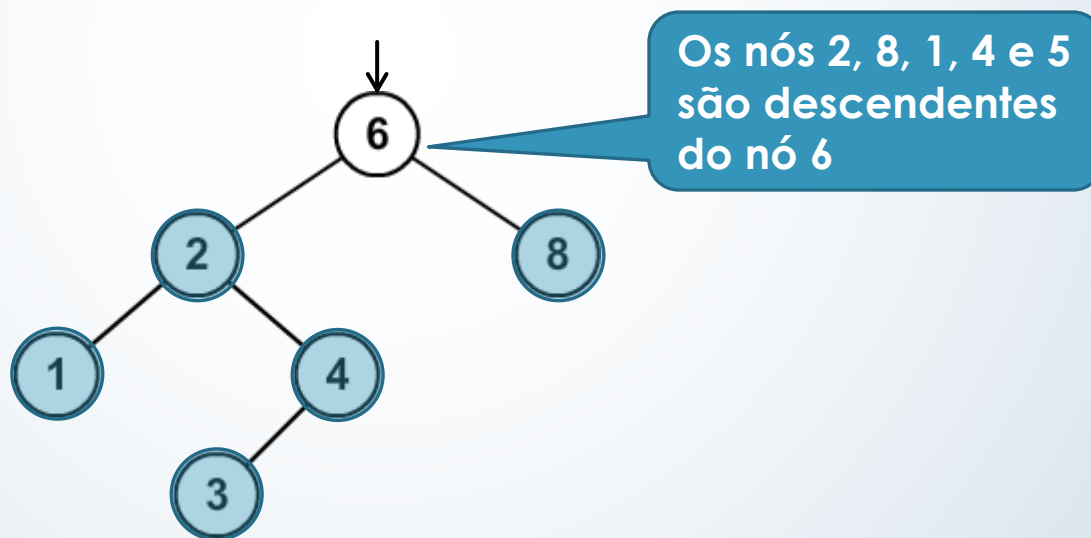
- **Nó ancestral:** nós que estão acima de um nó e possuem ligação direta ou indireta.



Árvore Binária

Propriedades

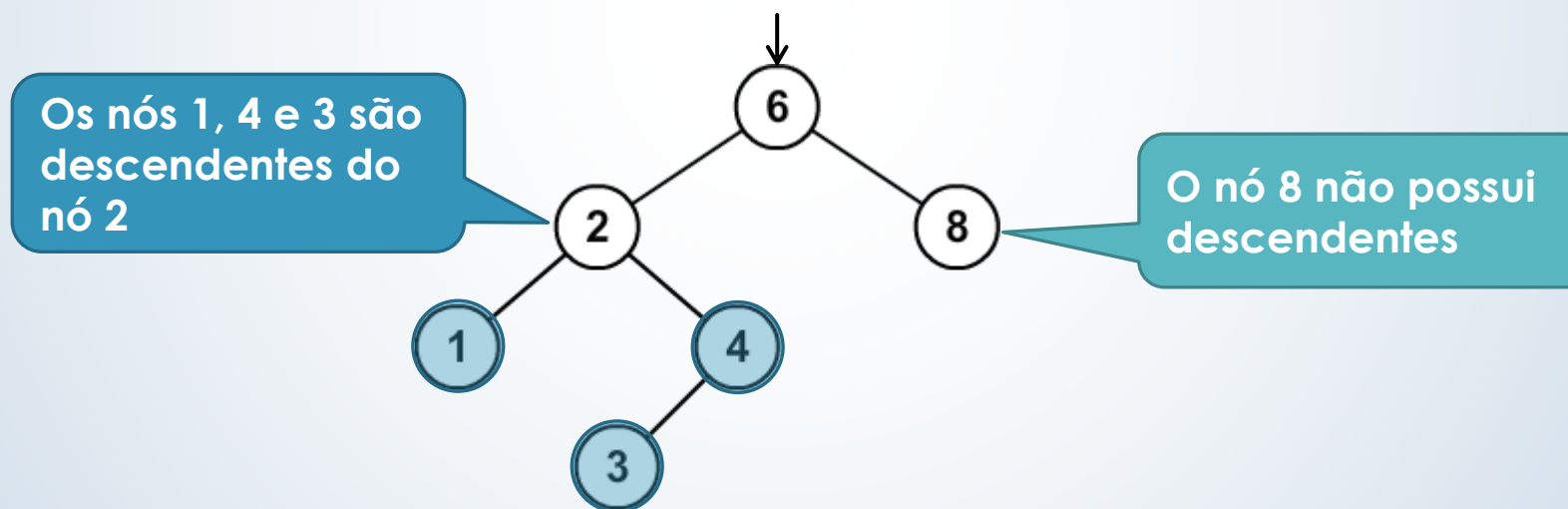
- **Nó descendente:** nós que estão abaixo de um nó e possuem ligação direta ou indireta.



Árvore Binária

Propriedades

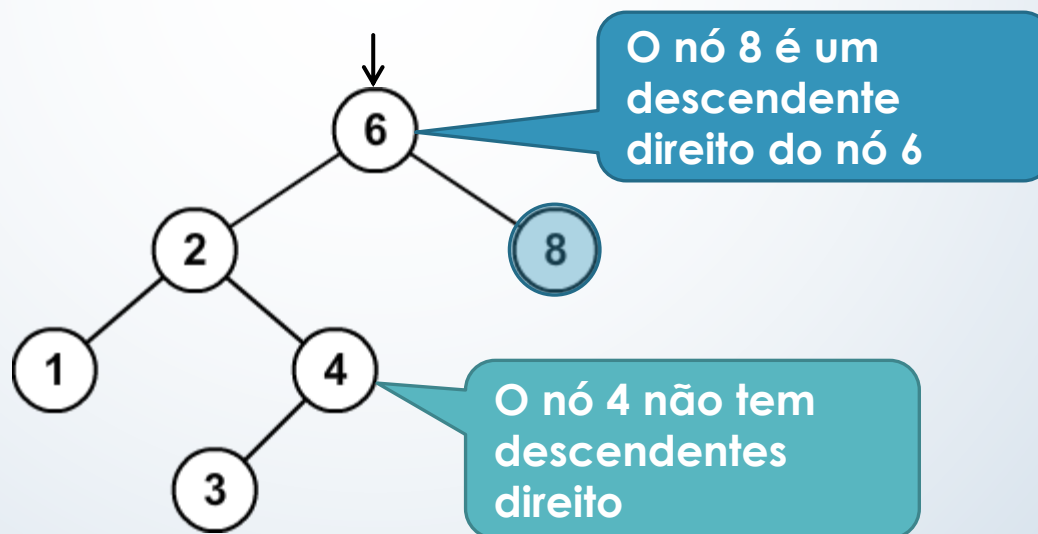
- **Nó descendente:** nós que estão abaixo de um nó e possuem ligação direta ou indireta.



Árvore Binária

Propriedades

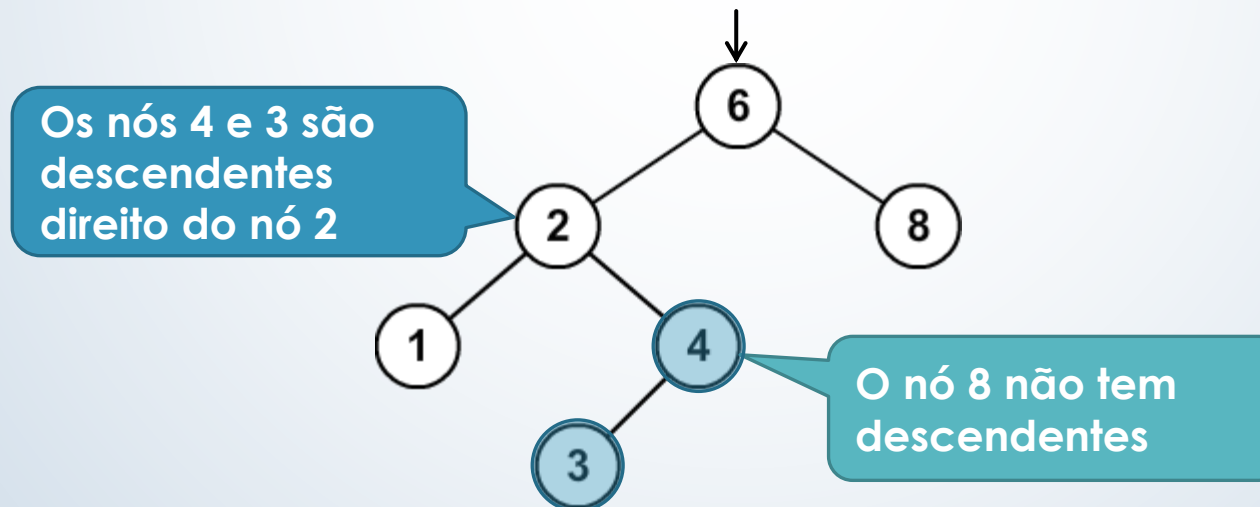
- **Nó descendente direito:** nós que estão abaixo de um nó, possuem ligação direta ou indireta e fazem parte da subárvore direita.



Árvore Binária

Propriedades

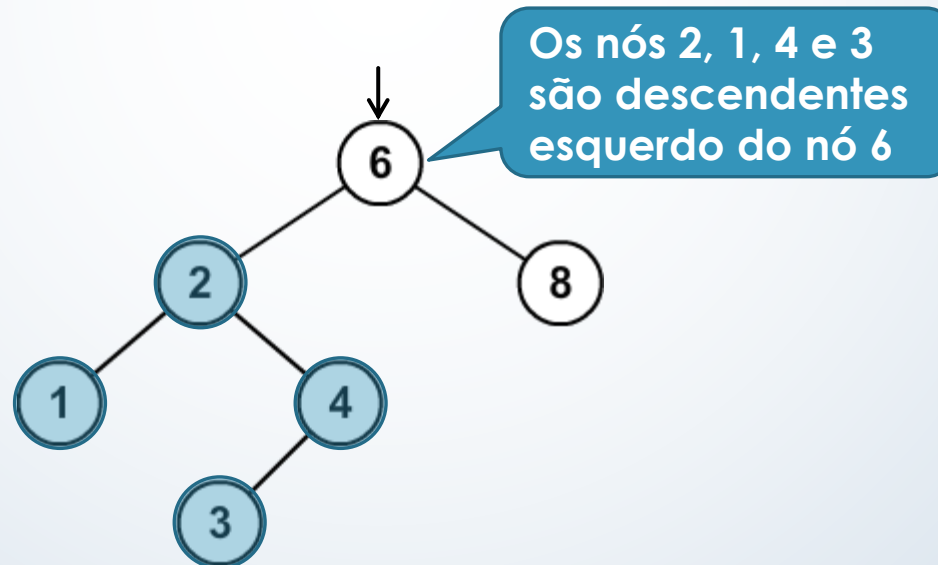
- **Nó descendente direito:** nós que estão abaixo de um nó, possuem ligação direta ou indireta e fazem parte da subárvore direita.



Árvore Binária

Propriedades

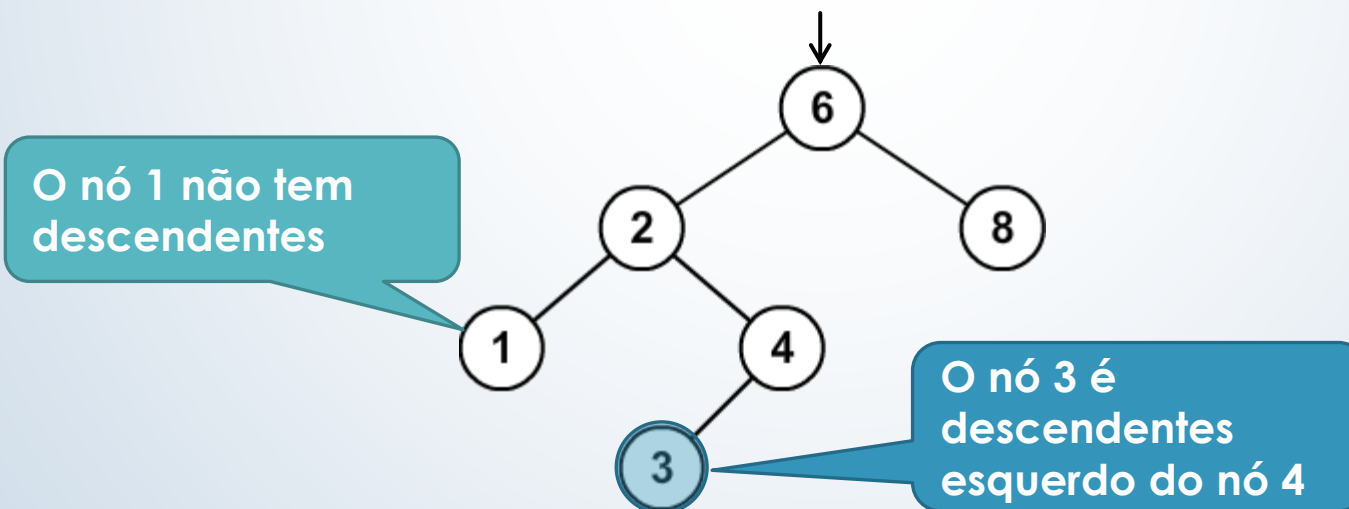
- **Nó descendente esquerdo:** nós que estão abaixo de um nó, possuem ligação direta ou indireta e fazem parte da subárvore esquerda.



Árvore Binária

Propriedades

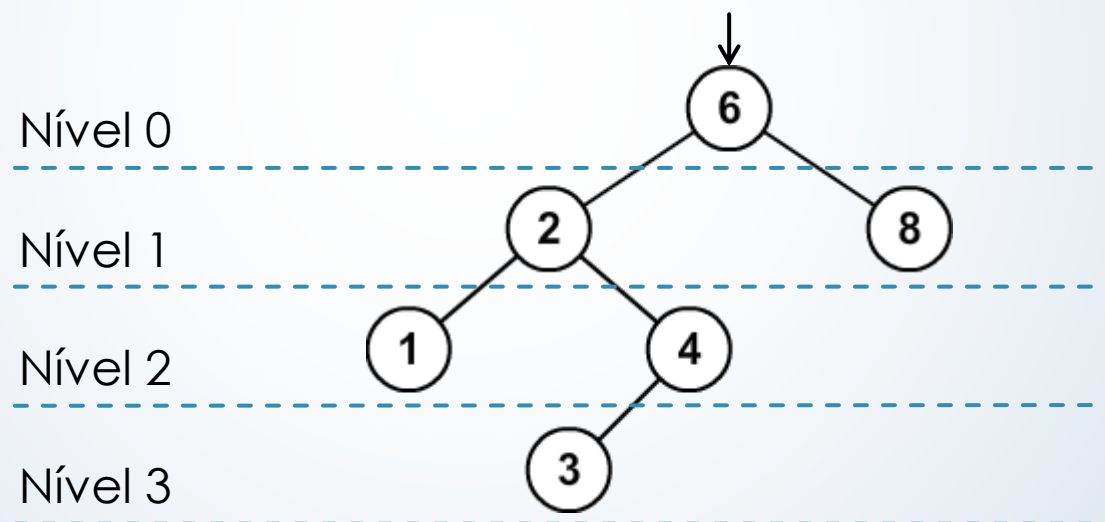
- **Nó descendente esquerdo:** nós que estão abaixo de um nó, possuem ligação direta ou indireta e fazem parte da subárvore esquerda.



Árvore Binária

Propriedades

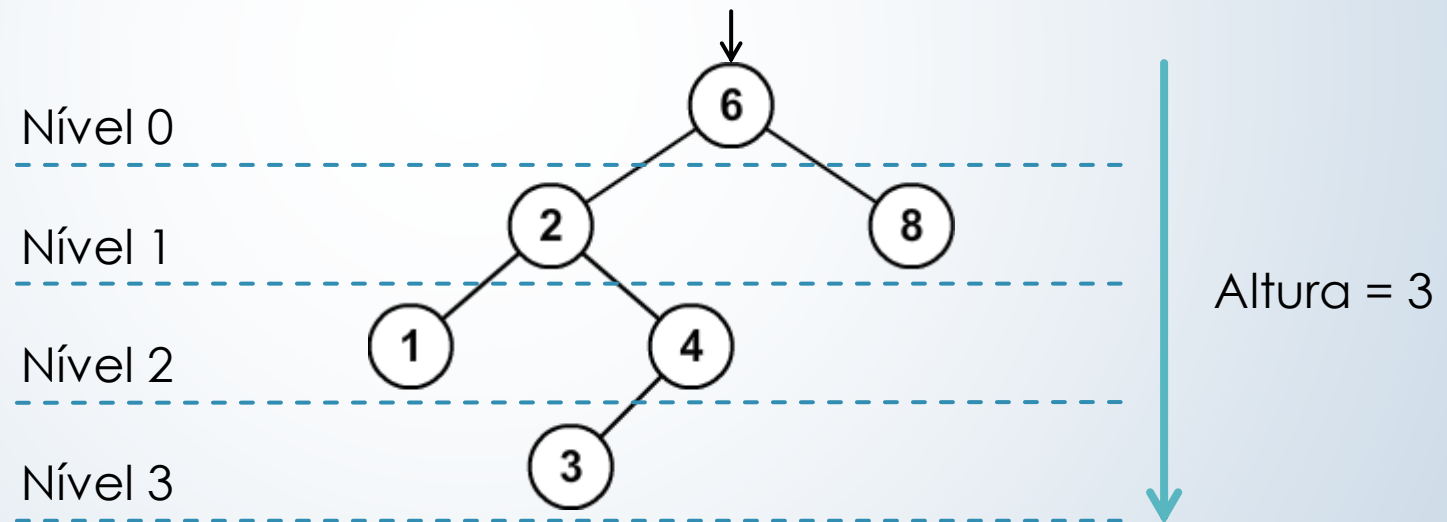
- **Nível de um nó:** é a sua distância em relação ao nó raiz. O nível da raiz é sempre zero.



Árvore Binária

Propriedades

- **Altura ou profundidade de uma árvore:** é o nível do nó mais distante da raiz.



Árvore Binária

Propriedades

► Número máximo de nós em um nível (k):

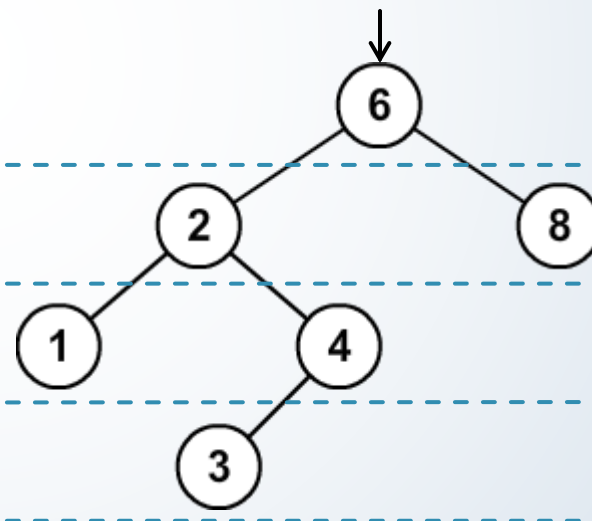
$$k = 2^n \mid n \text{ é o nível}$$

Nível 0 = $2^0 = 1$ nó

Nível 1 = $2^1 = 2$ nós

Nível 2 = $2^2 = 4$ nós

Nível 3 = $2^3 = 8$ nós

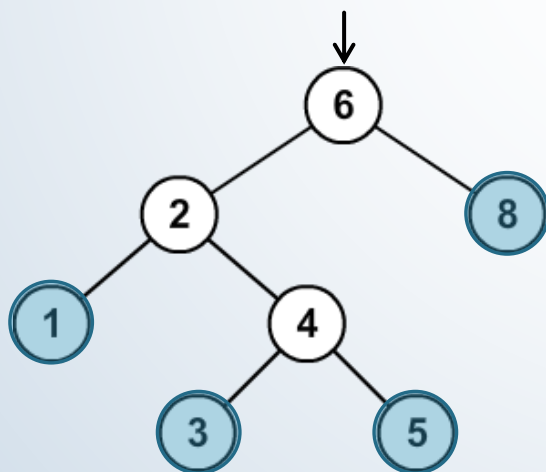


Árvore Binária

Propriedades

- **Árvore estritamente binária:** árvore em que todos os nós tem 0 ou 2 filhos;
- **Número de nós de uma árvore estritamente binária (k):**

$$k = 2n - 1 \mid n \text{ é a quantidade de nós folha}$$



Quantidade de nós folha: 4

Os nós folha são: 1, 3, 5, e 8

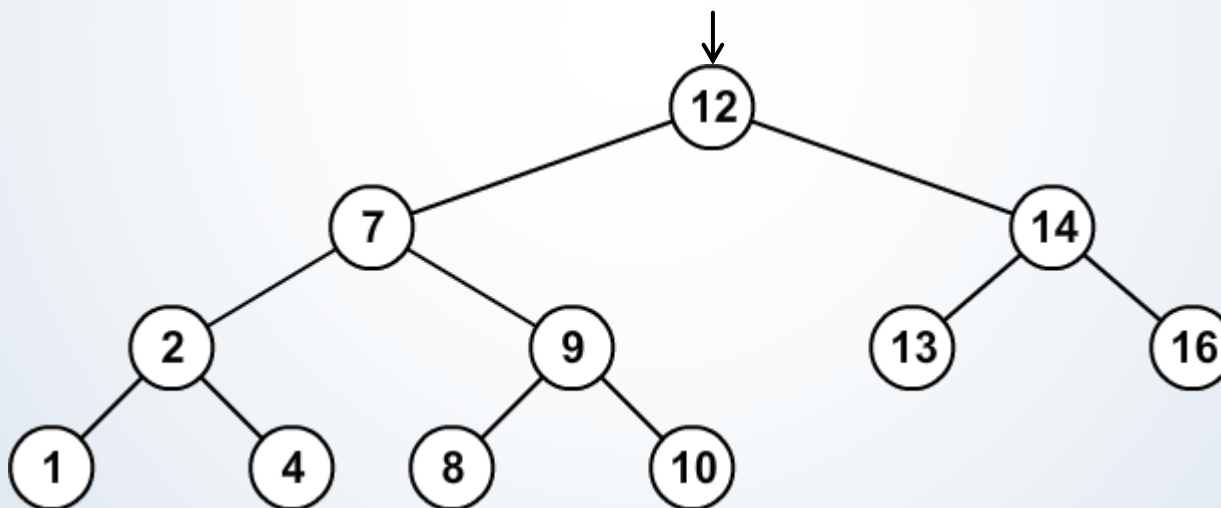
Número de nós da árvore estritamente binária:

$$k = 2n - 1 = 2(4) - 1 = 8 - 1 = 7$$

Árvore Binária

Propriedades

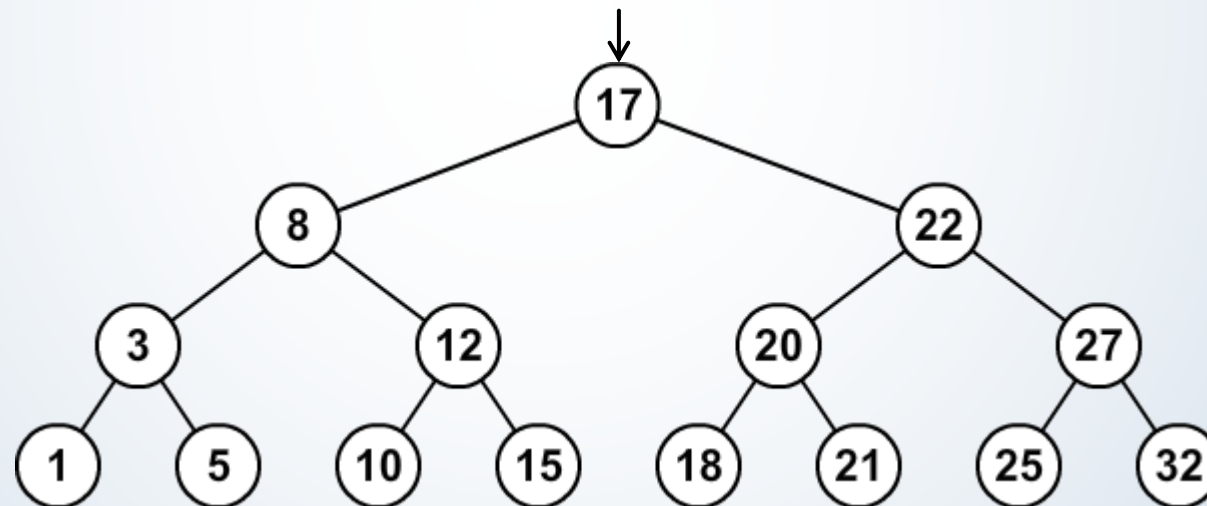
- **Árvore completa:** árvore binária em que todos os níveis, com exceção do último e do penúltimo, estão totalmente preenchidos e os nós sem filhos estão situados à esquerda.



Árvore Binária

Propriedades

- **Árvore cheia:** árvore binária em que se um nó tem alguma subárvore vazia, ele estará no último nível. É uma árvore estritamente binária e completa.



Árvore Binária de Busca

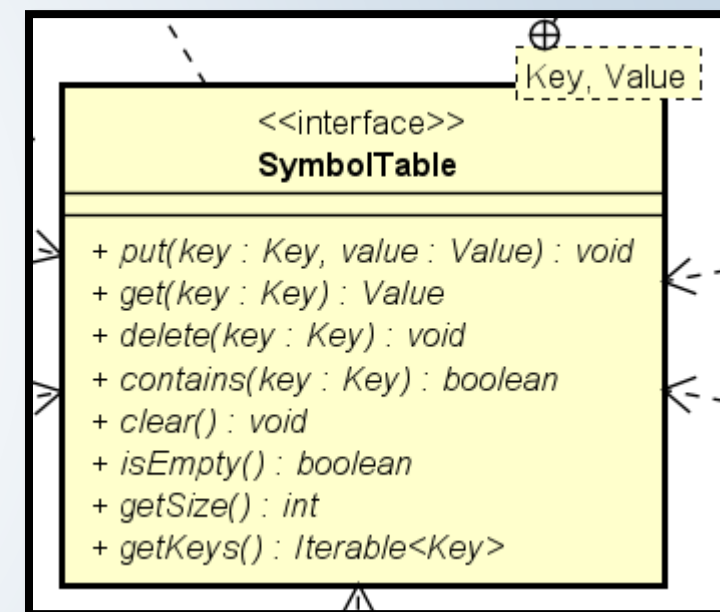
► Invariante/Condição de existência:

- Todos os nós de uma subárvore direita são maiores que o nó raiz;
- Todos os nós de uma subárvore esquerda são menores que o nó raiz;
- Cada subárvore é também uma árvore binária de busca.

Árvore Binária de Busca

► Implementação:

- Veremos agora a ideia por trás das operações de inserção (**put**) e remoção (**delete**) de nós de uma árvore binária de busca;
- Na nossa implementação real, contida na classe **BinarySearchTree**, as operações são feitas, em sua maioria, usando uma chave (**key**) como identificador do nó, tendo um valor associado à ela;
- As operações inerentes à nossa implementação real estão atreladas a API da tabela de símbolos;
- Nos próximos slides consideraremos, por questão de simplicidade (por enquanto), que a árvore armazena apenas as chaves.



Árvore Binária de Busca

Inserção (**put**)

raiz
↓
—

Árvore Binária de Busca

Inserção (**put**)

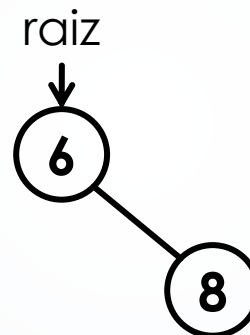
```
abb.put( 6 );
```



Árvore Binária de Busca

Inserção (**put**)

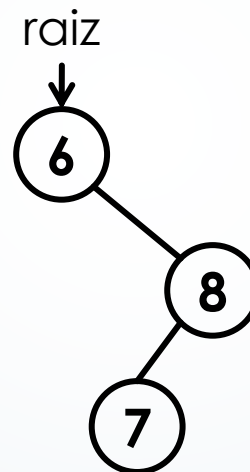
```
abb.put( 8 );
```



Árvore Binária de Busca

Inserção (**put**)

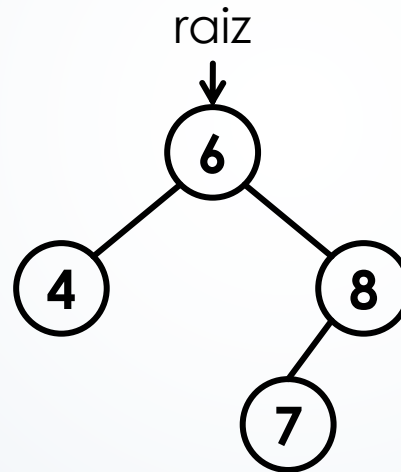
```
abb.put( 7 );
```



Árvore Binária de Busca

Inserção (**put**)

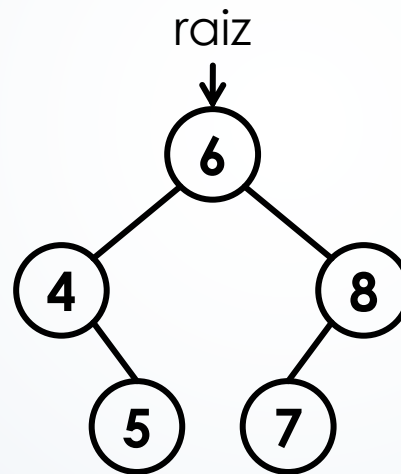
```
abb.put( 4 );
```



Árvore Binária de Busca

Inserção (**put**)

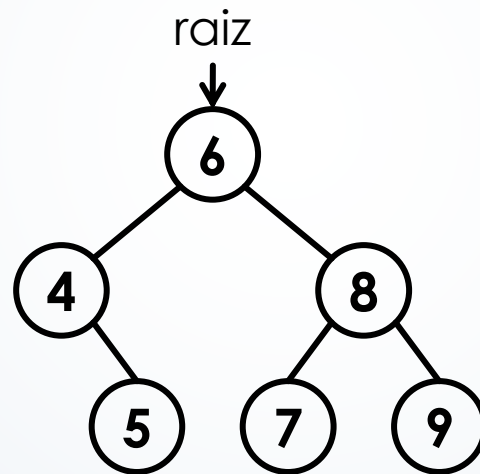
```
abb.put( 5 );
```



Árvore Binária de Busca

Inserção (**put**)

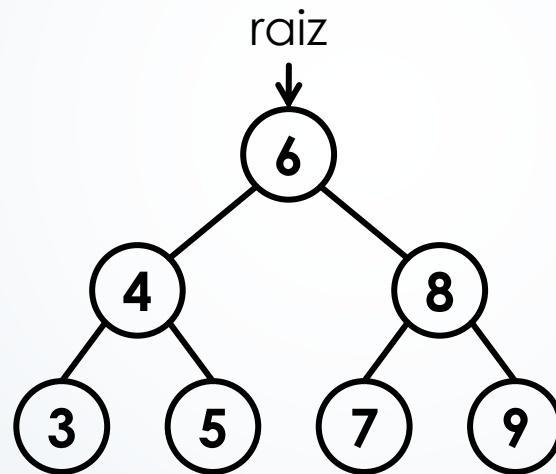
```
abb.put( 9 );
```



Árvore Binária de Busca

Inserção (**put**)

```
abb.put( 3 );
```



Árvore Binária de Busca

Remoção (**delete**)

➤ Se:

- Todos os nós de uma subárvore direita são maiores que o nó raiz;
- Todos os nós de uma subárvore esquerda são menores que o nó raiz;
- Cada subárvore é também uma árvore binária de busca;

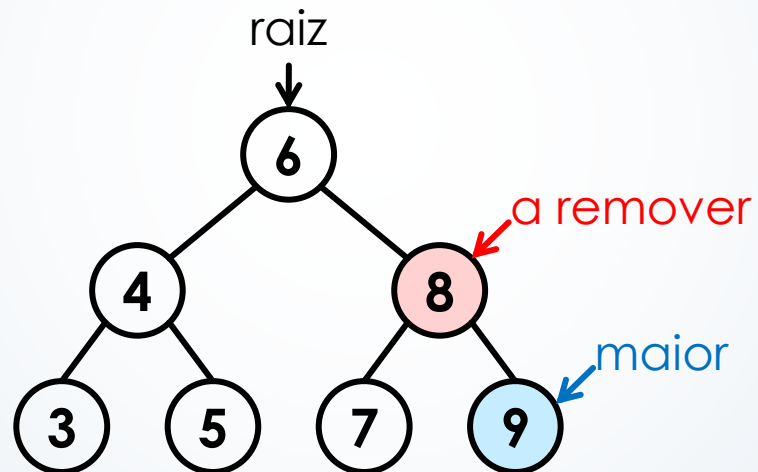
➤ Então:

- A remoção de um nó com filhos faz com que o nó maior que ele (à sua direita) ocupe sua posição;
- Se o nó à direita não existe, então o nó à esquerda ocupa sua posição;
- Caso o nó seja um nó folha, não há a necessidade de reestruturação da árvore;
- Essas condições são necessárias para manter a invariante das árvores binárias de busca. Note que pode-se realizar a mesma operação de forma reflexiva, ou seja, para o outro lado.

Árvore Binária de Busca

Remoção (**delete**)

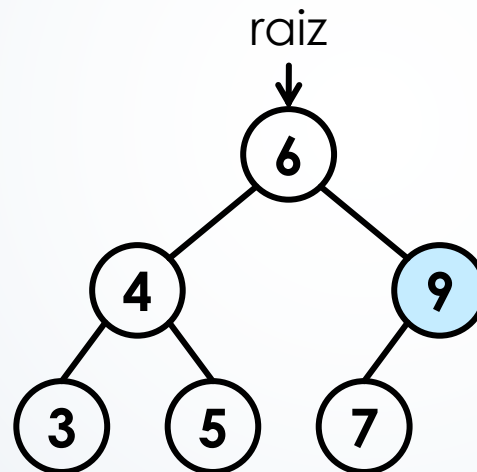
```
abb.delete( 8 );
```



Árvore Binária de Busca

Remoção (**delete**)

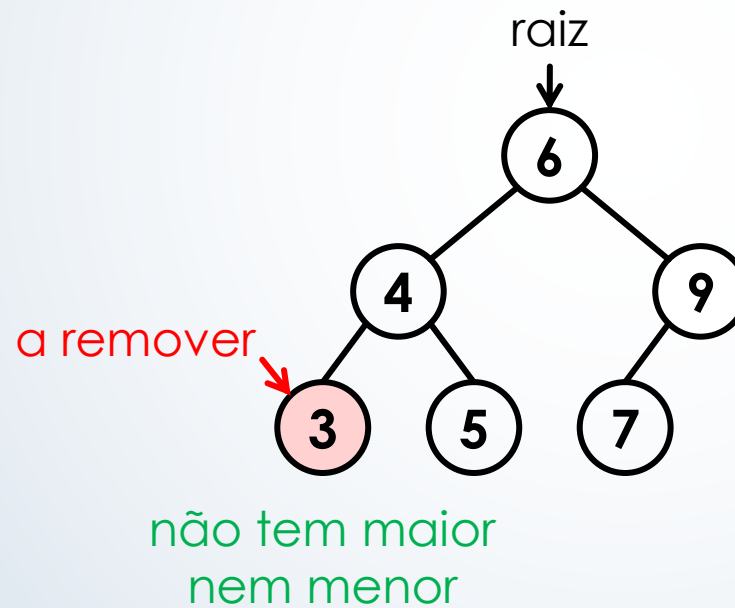
```
abb.delete( 8 );
```



Árvore Binária de Busca

Remoção (**delete**)

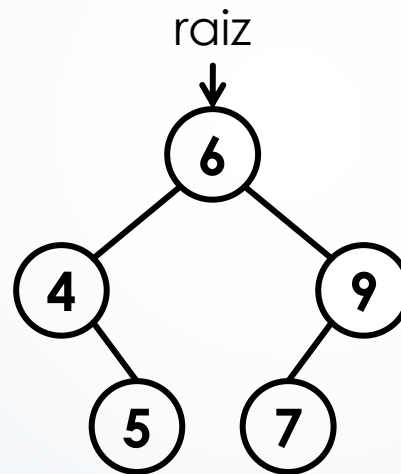
```
abb.delete( 3 );
```



Árvore Binária de Busca

Remoção (**delete**)

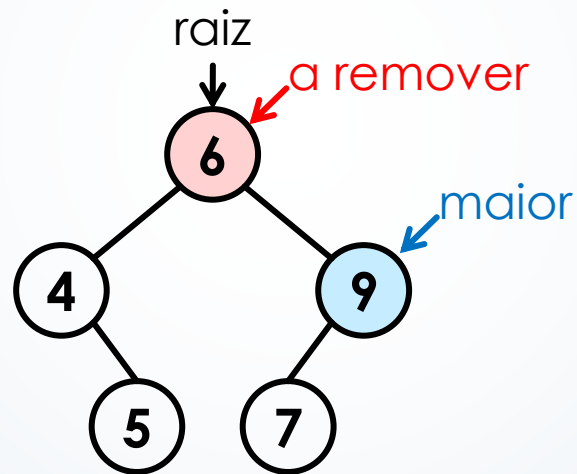
```
abb.delete( 3 );
```



Árvore Binária de Busca

Remoção (**delete**)

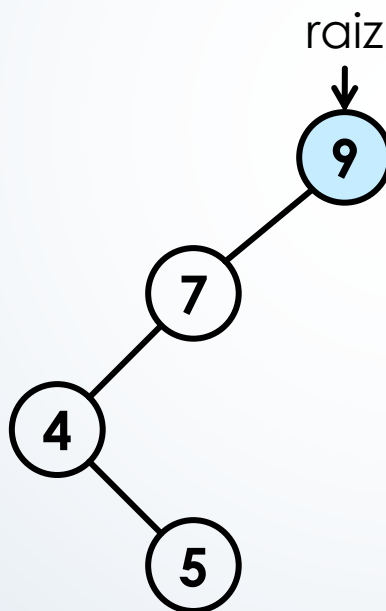
```
abb.delete( 6 );
```



Árvore Binária de Busca

Remoção (**delete**)

```
abb.delete( 6 );
```



Importante!

A política de remoção de nós, onde o maior nó ocupa a posição do nó excluído, faz com que, quando o nó removido possui filhos para os dois lados, a altura da árvore aumente! Esse algoritmo de remoção é denominado Ribbard Deletion.

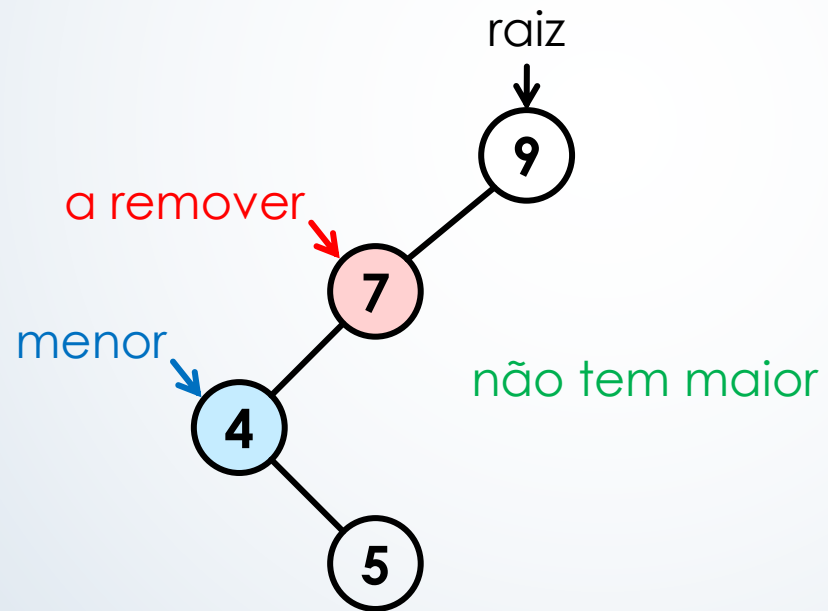
Para pensar:

- Isso é bom ou ruim?
- Há alguma situação análoga na inserção?

Árvore Binária de Busca

Remoção (**delete**)

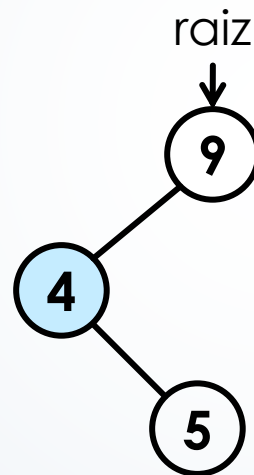
```
abb.delete( 7 );
```



Árvore Binária de Busca

Remoção (**delete**)

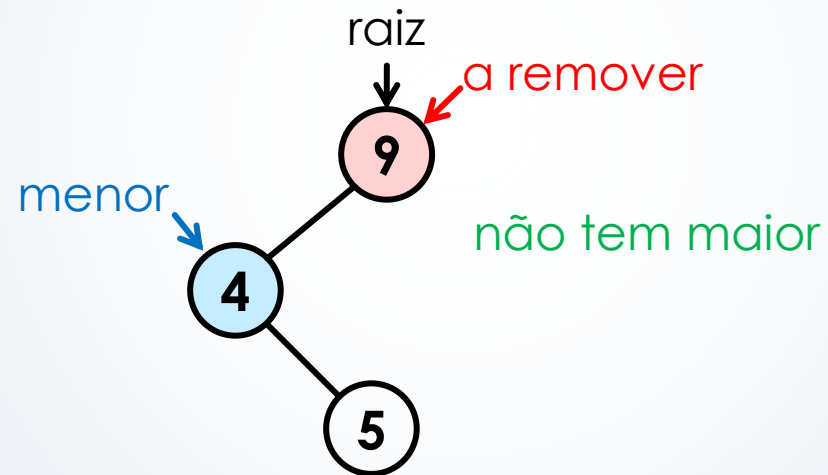
```
abb.delete( 7 );
```



Árvore Binária de Busca

Remoção (**delete**)

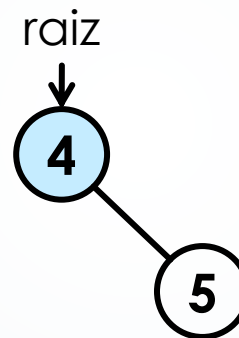
```
abb.delete( 9 );
```



Árvore Binária de Busca

Remoção (**delete**)

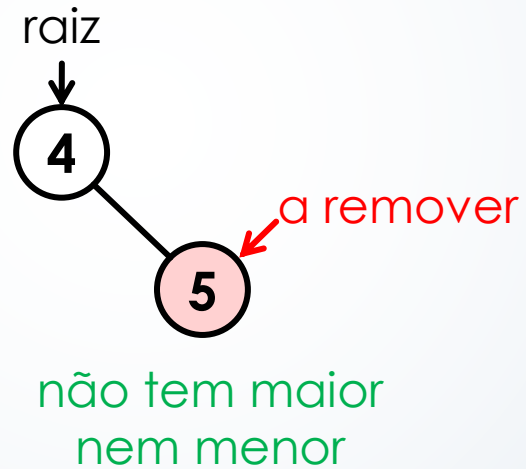
```
abb.delete( 9 );
```



Árvore Binária de Busca

Remoção (**delete**)

```
abb.delete( 5 );
```



Árvore Binária de Busca

Remoção (**delete**)

```
abb.delete( 5 );
```



Árvore Binária de Busca

Remoção (**delete**)

```
abb.delete( 4 );
```



Árvore Binária de Busca

Remoção (**delete**)

```
abb.delete( 4 );
```

raiz
↓
—

Árvore Binária

Para pensar

- As árvores binárias de busca, da forma que foram apresentadas até agora, tem um grave defeito. Você é capaz de identificá-lo? Pense... se uma árvore binária de busca será usada como infraestrutura para permitir que uma tabela de símbolos seja implementada, queremos sempre que as operações realizadas, como inserção, exclusão e consulta sejam o mais rápidas possíveis. Dada a natureza da construção das árvores binárias de busca, qual seria então essa falha?
- Não conseguiu? Construa uma árvore binária de busca com os seguintes valores, nessa exata ordem: 1, 2, 3, 4, 5, 6 e 7.
- O que você nota?
- Extrapole a ideia para a construção com mil elementos em ordem. Qual a ordem de crescimento da inserção, remoção e consulta no pior caso?

Árvore Binária

Percursos

- **Percurso:** forma de percorrer uma estrutura de dados, no caso, uma árvore binária;
- Principais tipos de percurso em árvores binárias:
 1. **Pré-ordem:** percorre-se primeiramente a subárvore esquerda, ao terminar processa-se a raiz e então percorre-se a subárvore direita. Chamado de **preorder traversal** em inglês;
 2. **Em ordem ou ordem simétrica:** primeiramente processa-se a raiz, depois percorre-se a subárvore esquerda e então percorre-se a subárvore direita. Chamado de **inorder traversal** em inglês;
 3. **Pós-ordem ou ordem final:** percorre-se primeiramente a subárvore direita, ao terminar processa-se a raiz e então percorre-se a subárvore esquerda. Chamado de **postorder traversal** em inglês;
 4. **Em nível:** iniciando no nível zero, progride-se nível a nível, processando cada raiz, da esquerda para a direita. Chamado de **level-order traversal** em inglês;

Árvore Binária

Percursos

➤ Observações:

- Os percursos pré-ordem, em ordem e pós-ordem são análogos à Busca em Profundidade (**Depth-First Search**) em grafos;
- O percurso em nível é análogo à Busca em Largura (**Breadth-First Search**) em grafos;

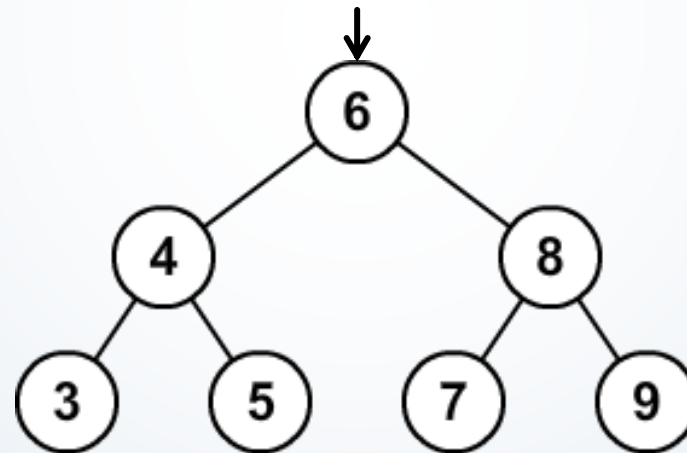
➤ Aplicações:

1. **Pré-ordem:** cópia de uma árvore binária de busca, criação de hierarquias, como índices em documentos de texto, ordenação topológica (grafos), obtenção de expressão pré-fixada em uma árvore de expressão etc;
2. **Em ordem:** processa os nós de forma crescente, podendo ser empregado, por exemplo, em ordenações, obtenção de expressão infixada em uma árvore de expressão etc;
3. **Pós-ordem:** ordenação topológica (grafos), obtenção de expressão pós-fixada em uma árvore de expressão etc;
4. **Em nível:** obtenção da distância entre nós.

Árvore Binária

Percursos – Pré-ordem

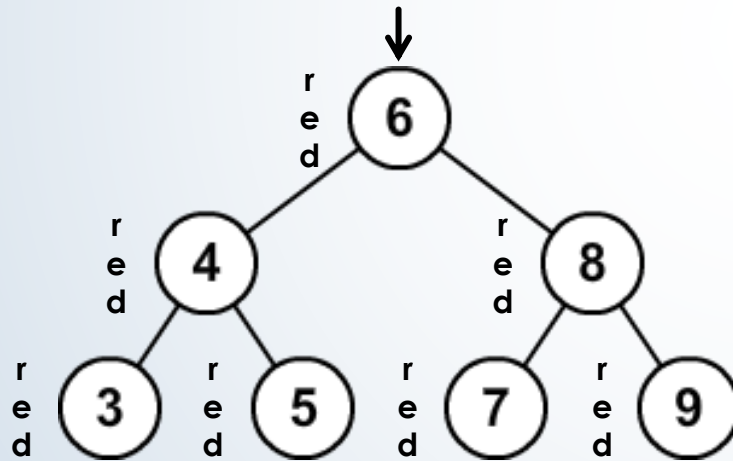
➡ Raiz ➡ Esquerda ➡ Direita



Árvore Binária

Percursos – Pré-ordem

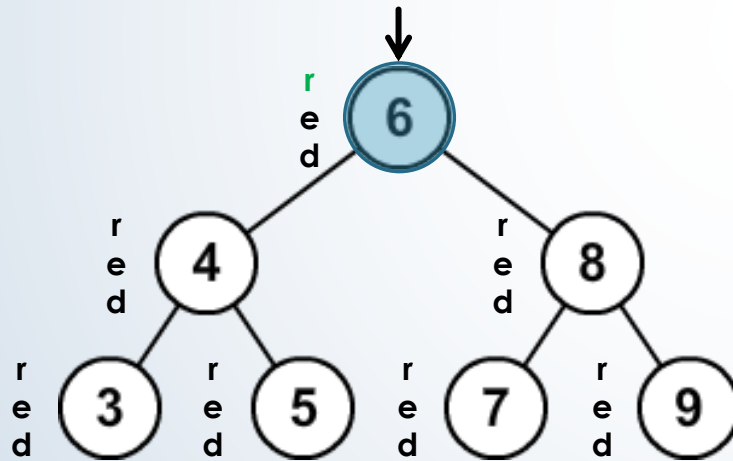
➡ Raiz ➡ Esquerda ➡ Direita



Árvore Binária

Percursos – Pré-ordem

➡ Raiz ➡ Esquerda ➡ Direita

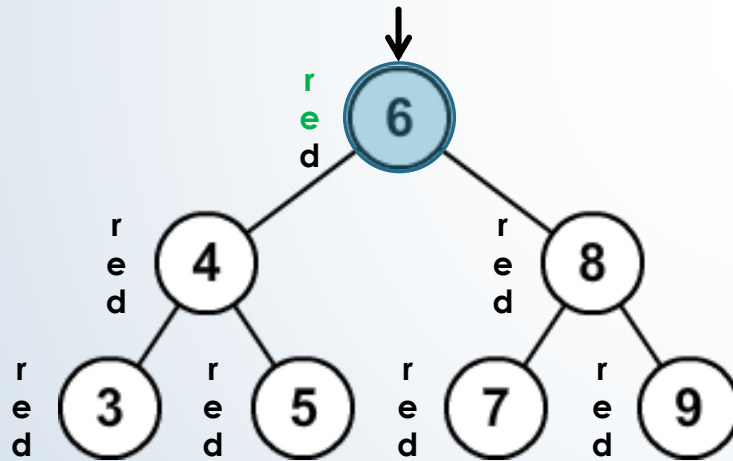


6

Árvore Binária

Percursos – Pré-ordem

➡ Raiz ➡ Esquerda ➡ Direita

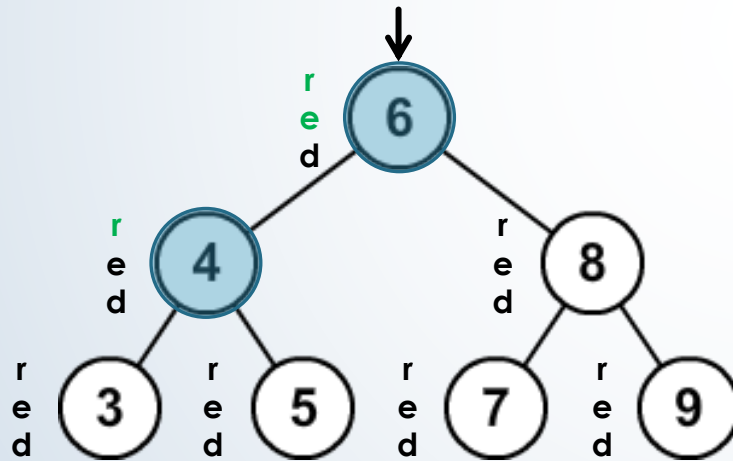


6

Árvore Binária

Percursos – Pré-ordem

➡ Raiz ➡ Esquerda ➡ Direita

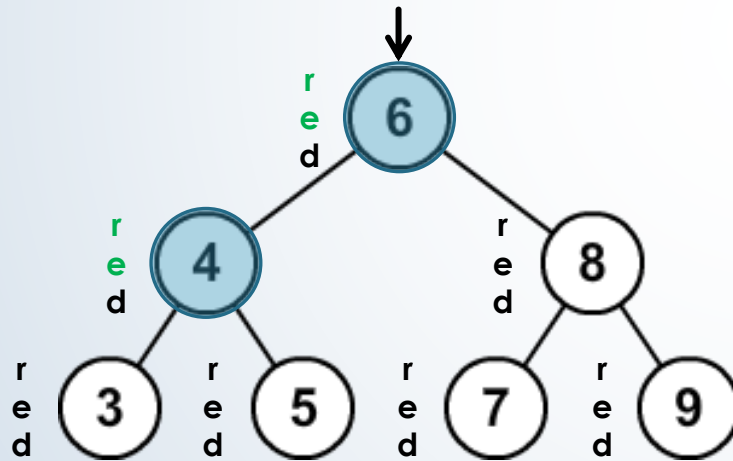


6 4

Árvore Binária

Percursos – Pré-ordem

➡ Raiz ➡ Esquerda ➡ Direita

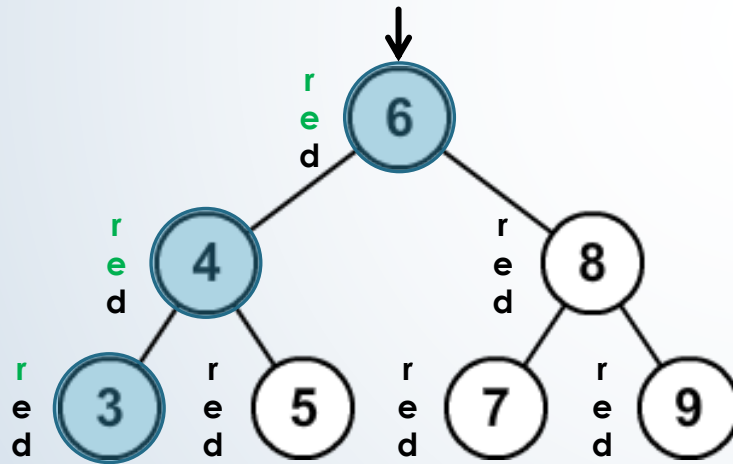


6 4

Árvore Binária

Percursos – Pré-ordem

➡ Raiz ➡ Esquerda ➡ Direita

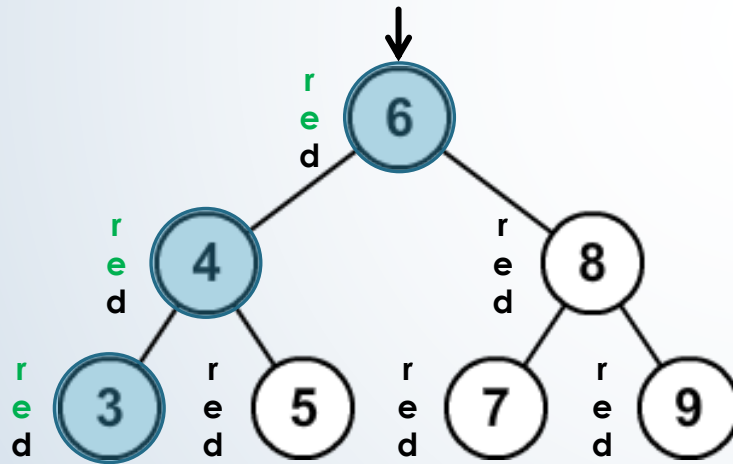


6 4 3

Árvore Binária

Percursos – Pré-ordem

➡ Raiz ➡ Esquerda ➡ Direita

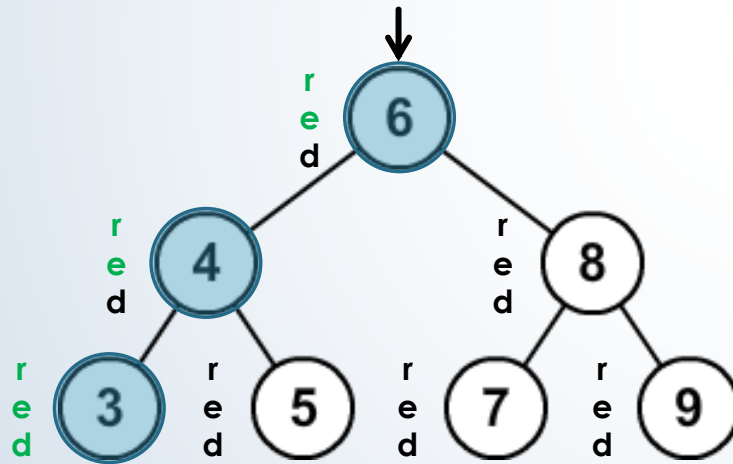


6 4 3

Árvore Binária

Percursos – Pré-ordem

➡ Raiz ➡ Esquerda ➡ Direita

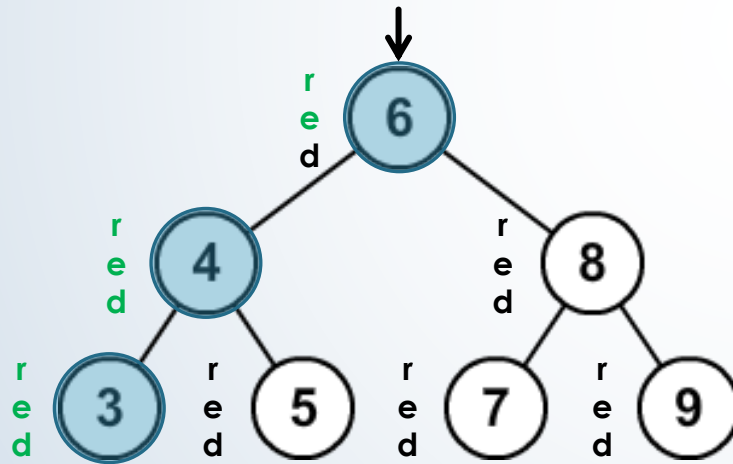


6 4 3

Árvore Binária

Percursos – Pré-ordem

➡ Raiz ➡ Esquerda ➡ Direita

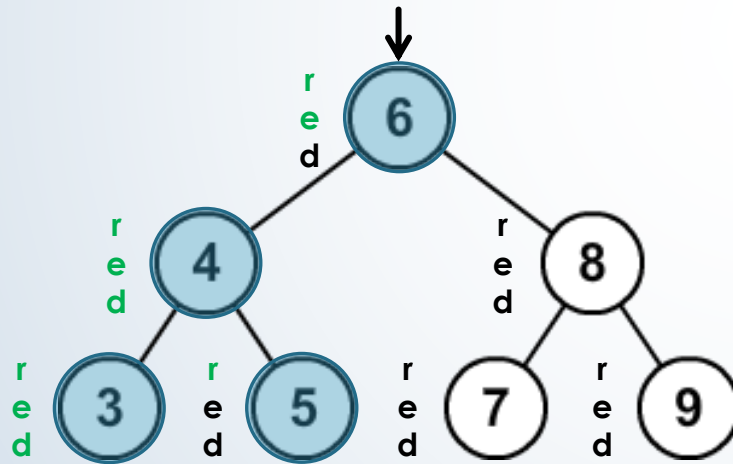


6 4 3

Árvore Binária

Percursos – Pré-ordem

➡ Raiz ➡ Esquerda ➡ Direita

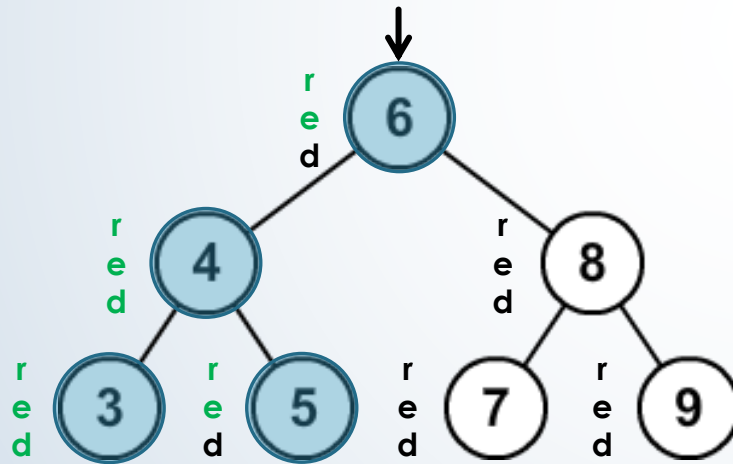


6 4 3 5

Árvore Binária

Percursos – Pré-ordem

➡ Raiz ➡ Esquerda ➡ Direita

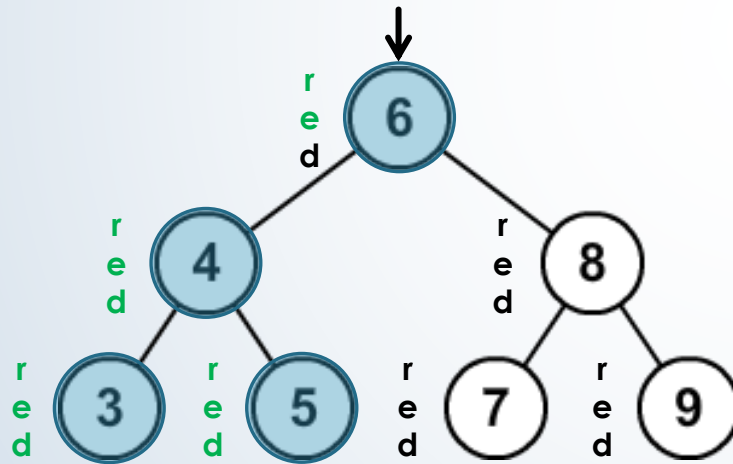


6 4 3 5

Árvore Binária

Percursos – Pré-ordem

➡ Raiz ➡ Esquerda ➡ Direita

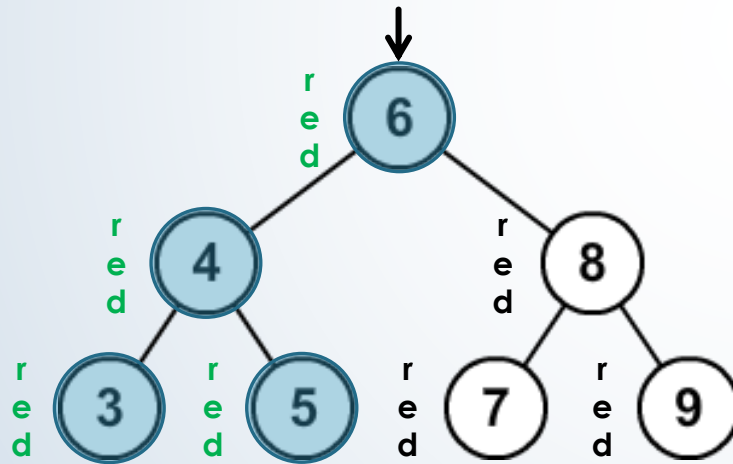


6 4 3 5

Árvore Binária

Percursos – Pré-ordem

➡ Raiz ➡ Esquerda ➡ Direita

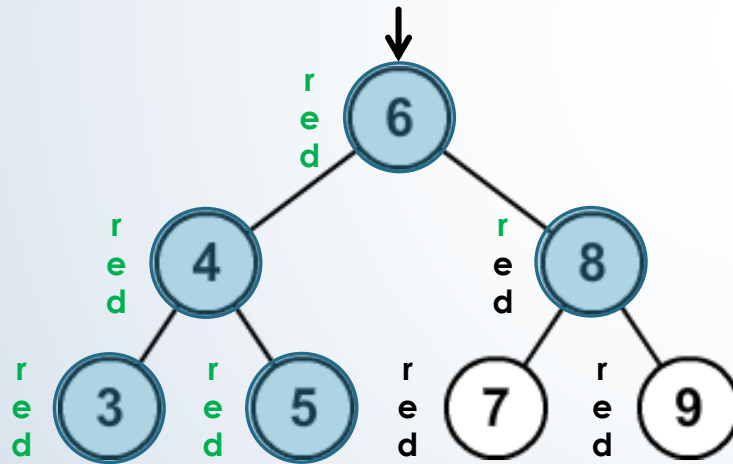


6 4 3 5

Árvore Binária

Percursos – Pré-ordem

➡ Raiz ➡ Esquerda ➡ Direita

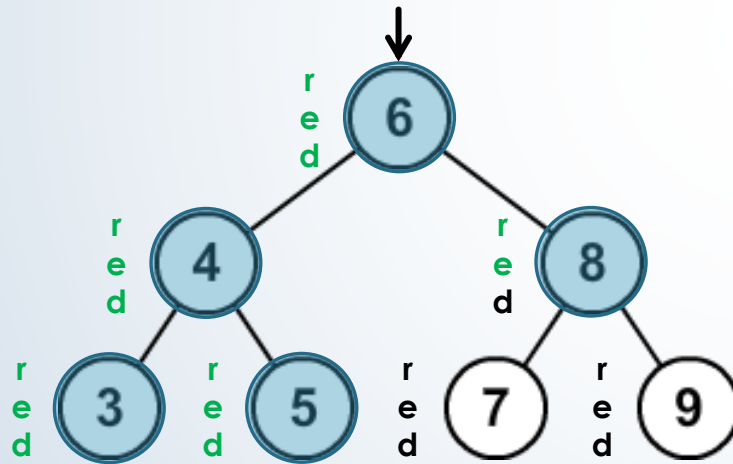


6 4 3 5 8

Árvore Binária

Percursos – Pré-ordem

➡ Raiz ➡ Esquerda ➡ Direita

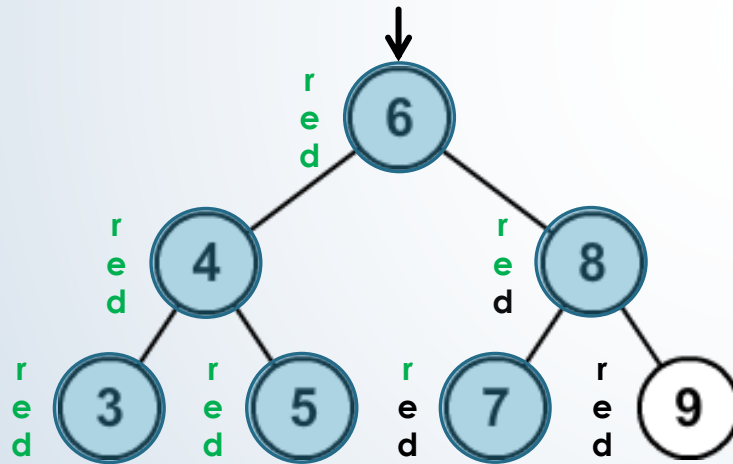


6 4 3 5 8

Árvore Binária

Percursos – Pré-ordem

➡ Raiz ➡ Esquerda ➡ Direita

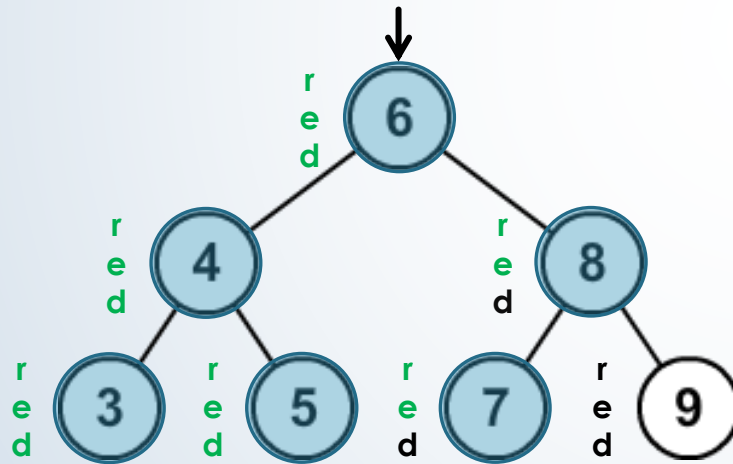


6 4 3 5 8 7

Árvore Binária

Percursos – Pré-ordem

➡ Raiz ➡ Esquerda ➡ Direita

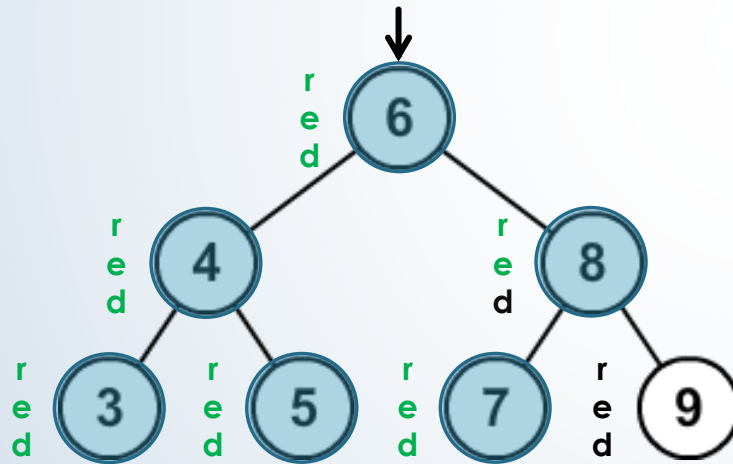


6 4 3 5 8 7

Árvore Binária

Percursos – Pré-ordem

➡ Raiz ➡ Esquerda ➡ Direita

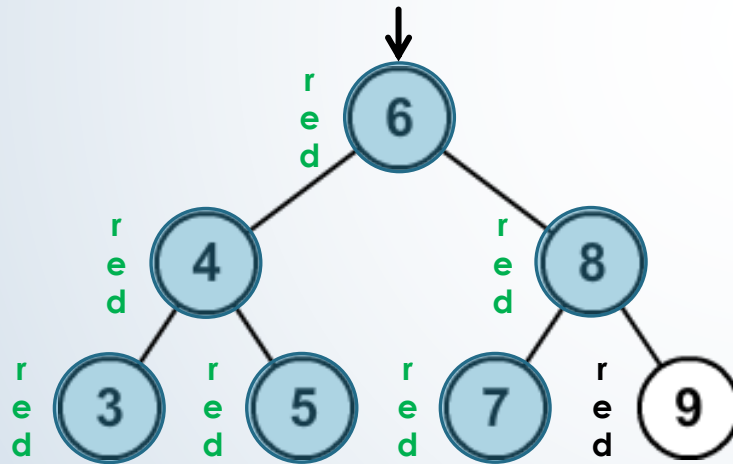


6 4 3 5 8 7

Árvore Binária

Percursos – Pré-ordem

➡ Raiz ➡ Esquerda ➡ Direita

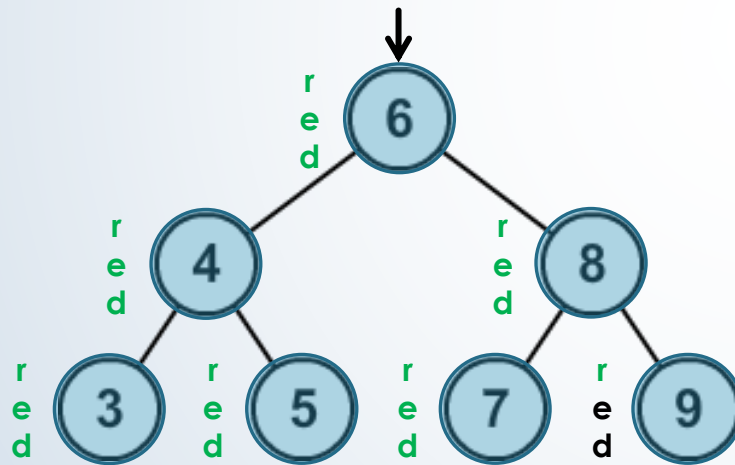


6 4 3 5 8 7

Árvore Binária

Percursos – Pré-ordem

➡ Raiz ➡ Esquerda ➡ Direita

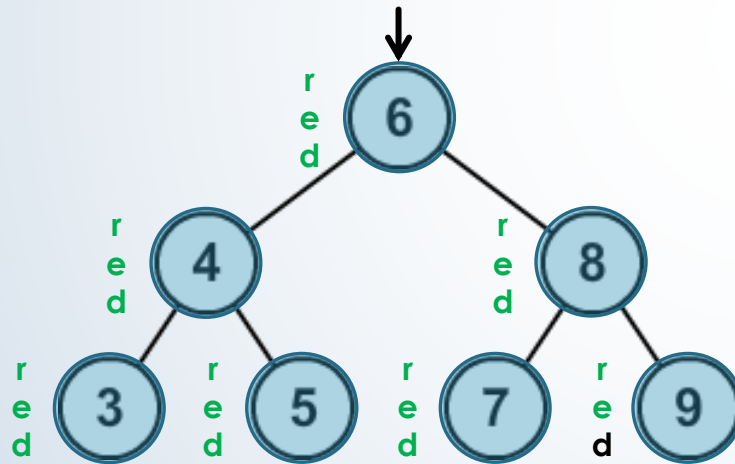


6 4 3 5 8 7 9

Árvore Binária

Percursos – Pré-ordem

➡ Raiz ➡ Esquerda ➡ Direita

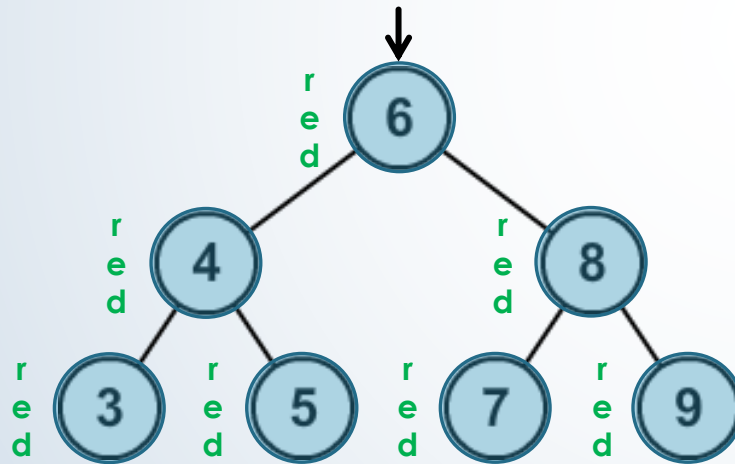


6 4 3 5 8 7 9

Árvore Binária

Percursos – Pré-ordem

➡ Raiz ➡ Esquerda ➡ Direita

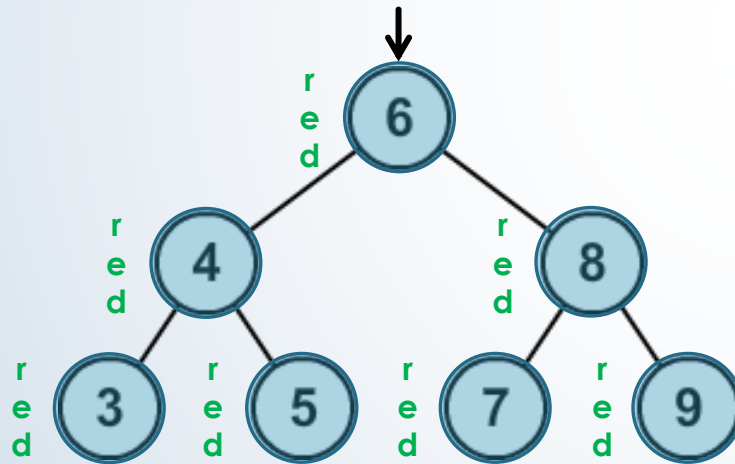


6 4 3 5 8 7 9

Árvore Binária

Percursos – Pré-ordem

➡ Raiz ➡ Esquerda ➡ Direita

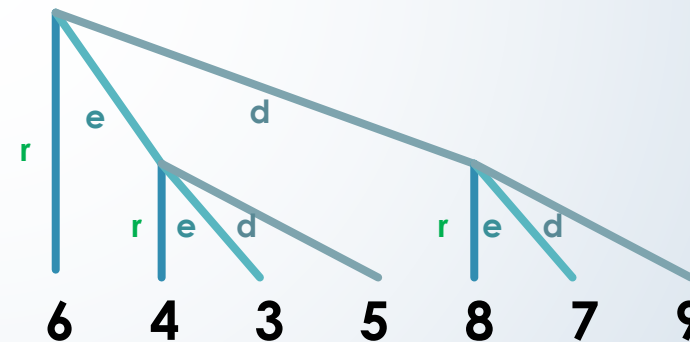
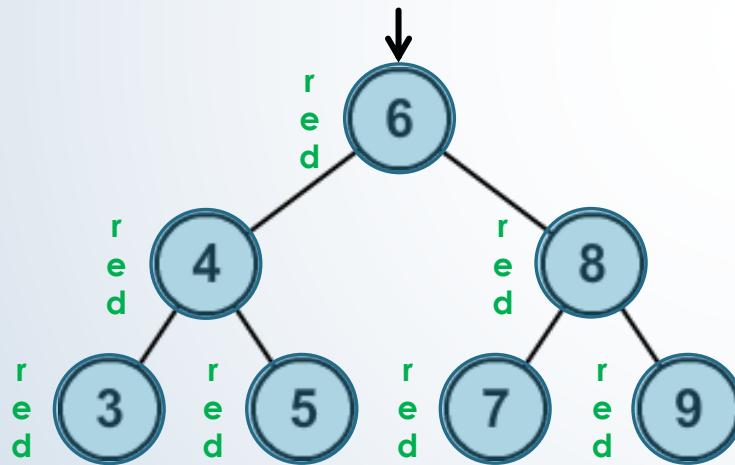


6 4 3 5 8 7 9

Árvore Binária

Percursos – Pré-ordem

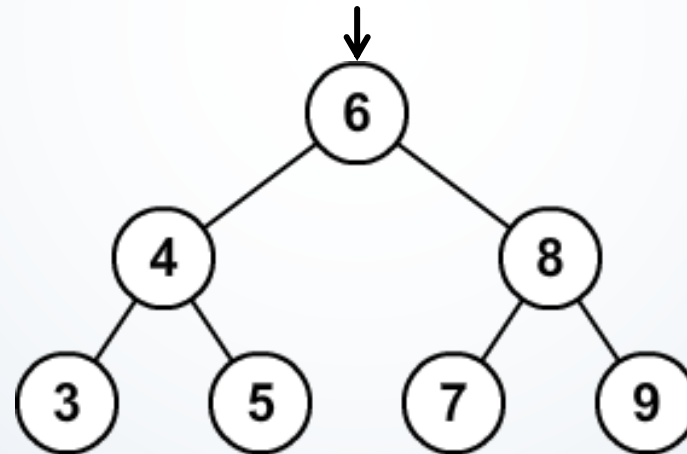
➡ Raiz ➡ Esquerda ➡ Direita



Árvore Binária

Percursos – Em ordem (ordem simétrica)

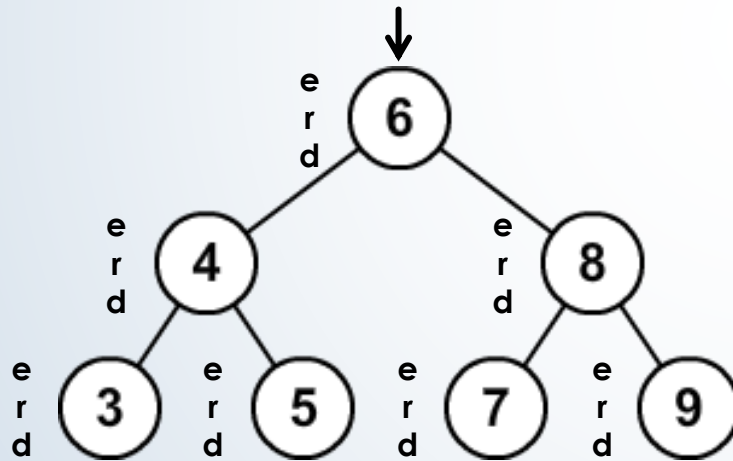
➡ Esquerda ➔ Raiz ➔ Direita



Árvore Binária

Percursos – Em ordem (ordem simétrica)

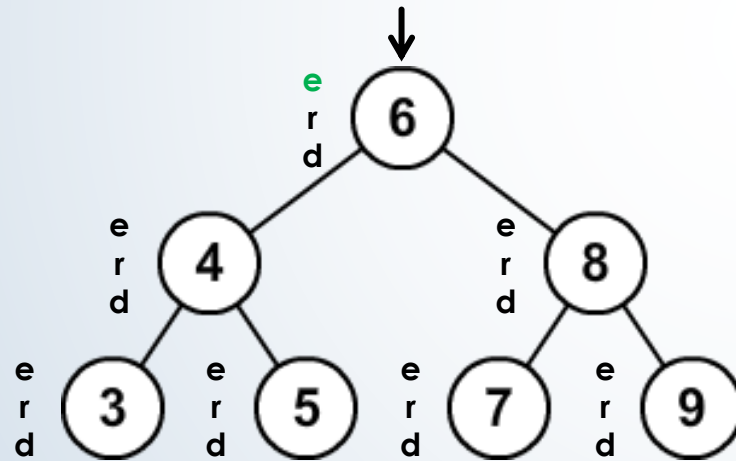
➡ Esquerda ➔ Raiz ➔ Direita



Árvore Binária

Percursos – Em ordem (ordem simétrica)

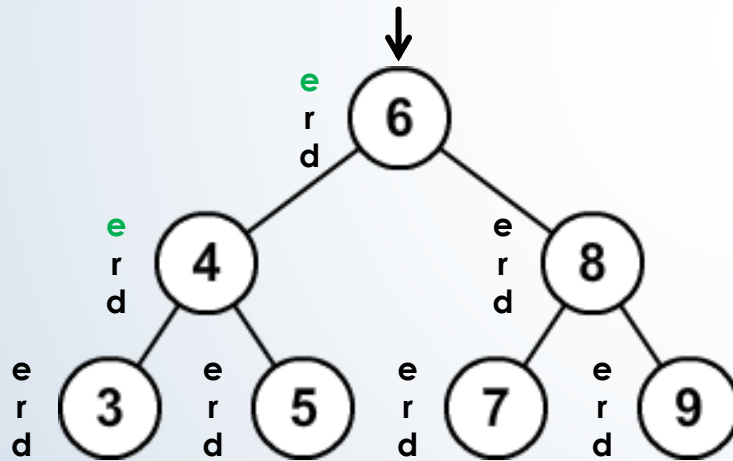
➡ Esquerda ➔ Raiz ➔ Direita



Árvore Binária

Percursos – Em ordem (ordem simétrica)

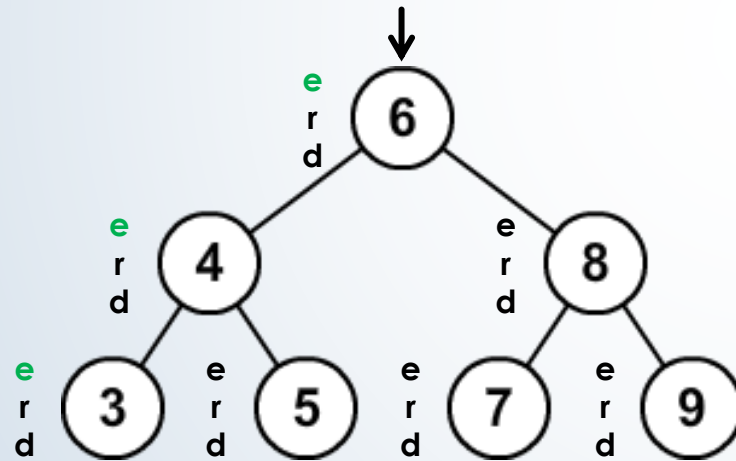
➡ Esquerda ➔ Raiz ➔ Direita



Árvore Binária

Percursos – Em ordem (ordem simétrica)

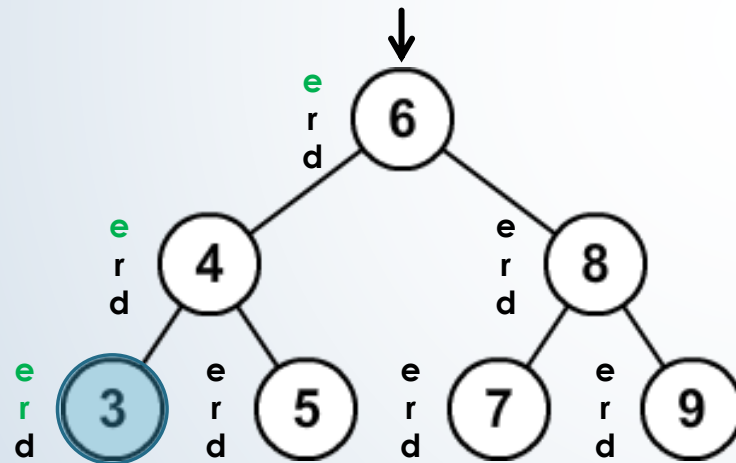
➡ Esquerda ➡ Raiz ➡ Direita



Árvore Binária

Percursos – Em ordem (ordem simétrica)

➡ Esquerda ➔ Raiz ➔ Direita

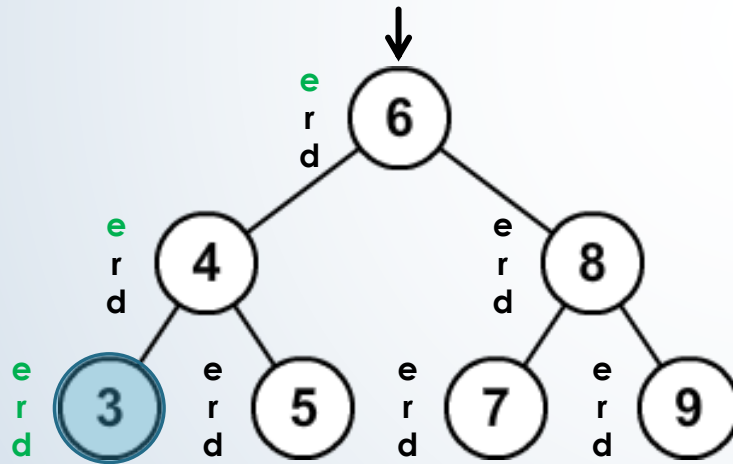


3

Árvore Binária

Percursos – Em ordem (ordem simétrica)

➡ Esquerda ➔ Raiz ➔ Direita

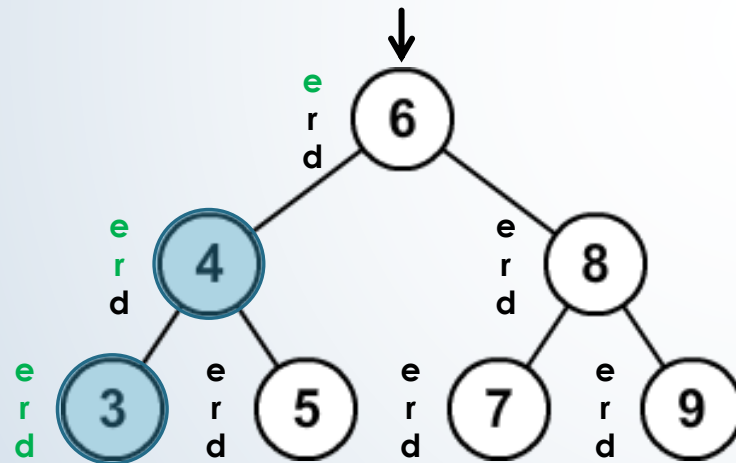


3

Árvore Binária

Percursos – Em ordem (ordem simétrica)

➡ Esquerda ➔ Raiz ➔ Direita

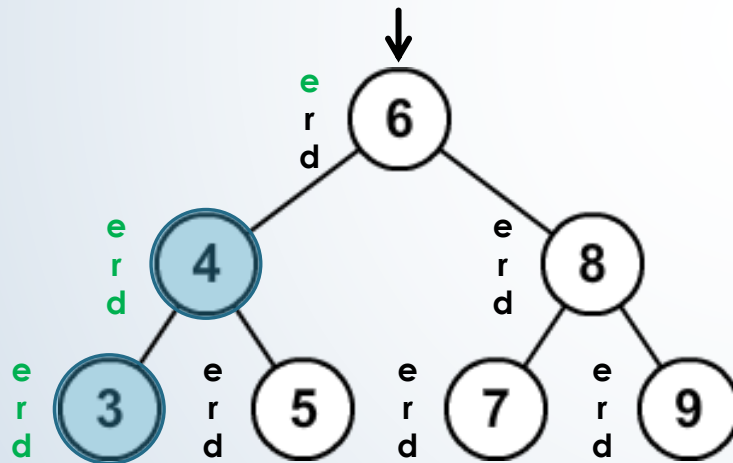


3 4

Árvore Binária

Percursos – Em ordem (ordem simétrica)

➡ Esquerda ➔ Raiz ➔ Direita

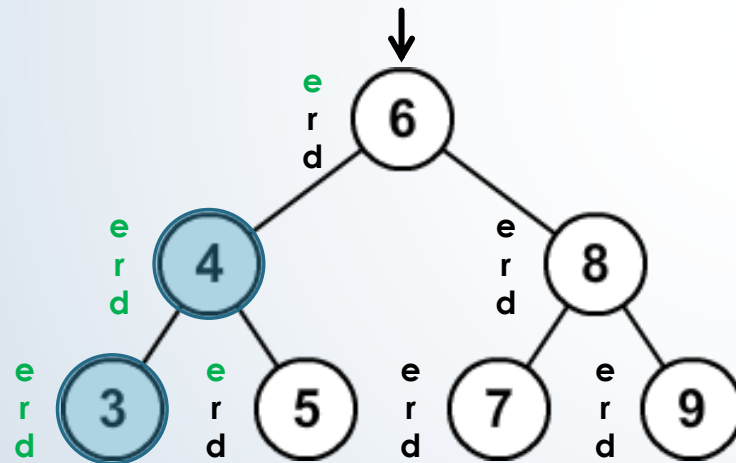


3 4

Árvore Binária

Percursos – Em ordem (ordem simétrica)

➡ Esquerda ➔ Raiz ➔ Direita

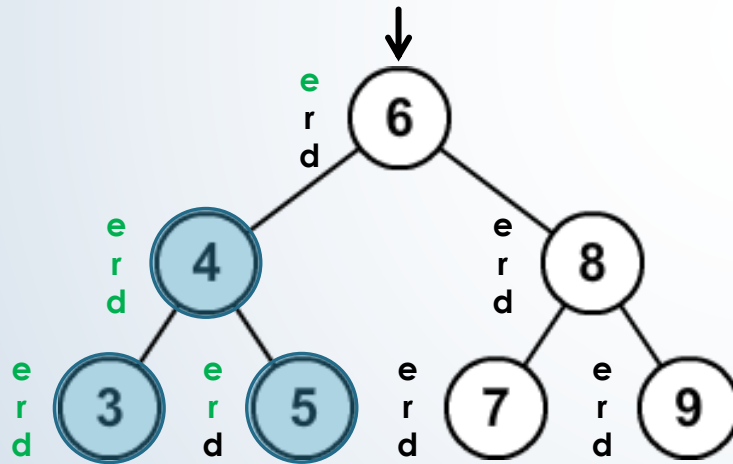


3 4

Árvore Binária

Percursos – Em ordem (ordem simétrica)

➡ Esquerda ➔ Raiz ➔ Direita

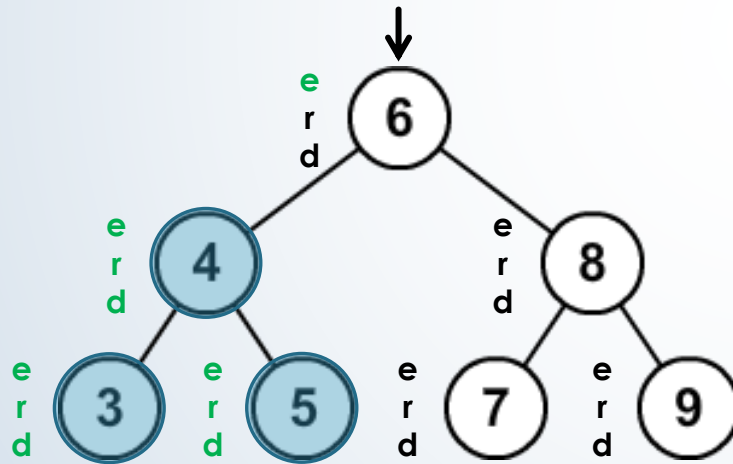


3 4 5

Árvore Binária

Percursos – Em ordem (ordem simétrica)

➡ Esquerda ➔ Raiz ➔ Direita

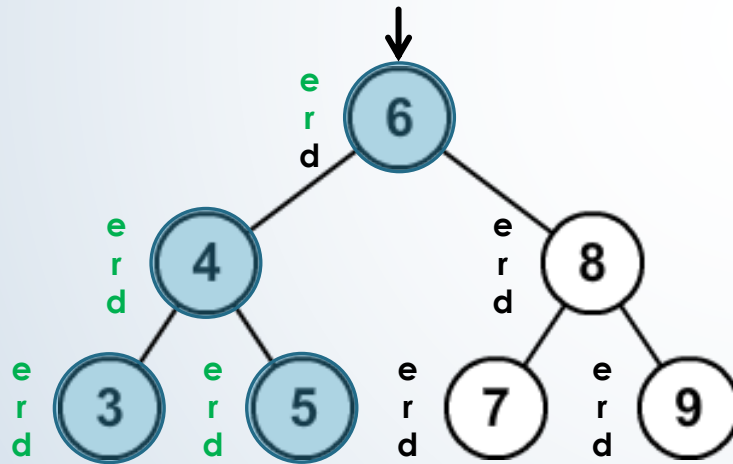


3 4 5

Árvore Binária

Percursos – Em ordem (ordem simétrica)

➡ Esquerda ➔ Raiz ➔ Direita

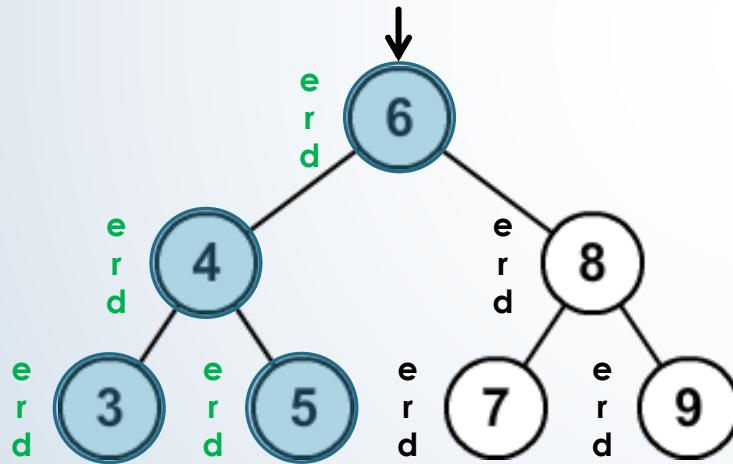


3 4 5 6

Árvore Binária

Percursos – Em ordem (ordem simétrica)

➡ Esquerda ➔ Raiz ➔ Direita

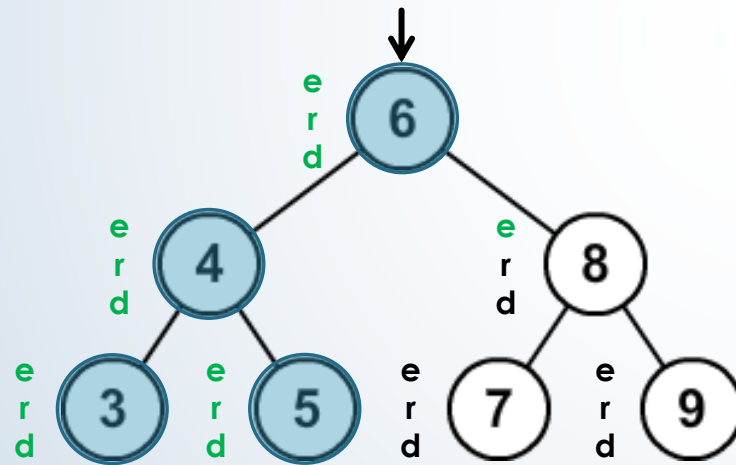


3 4 5 6

Árvore Binária

Percursos – Em ordem (ordem simétrica)

➡ Esquerda ➔ Raiz ➔ Direita

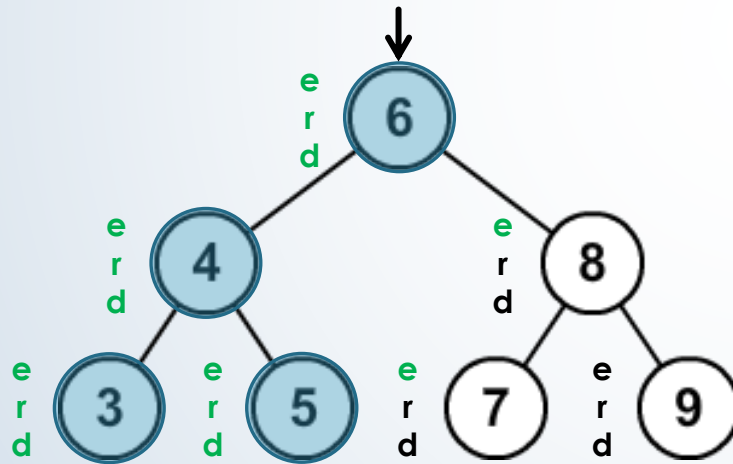


3 4 5 6

Árvore Binária

Percursos – Em ordem (ordem simétrica)

➡ Esquerda ➔ Raiz ➔ Direita

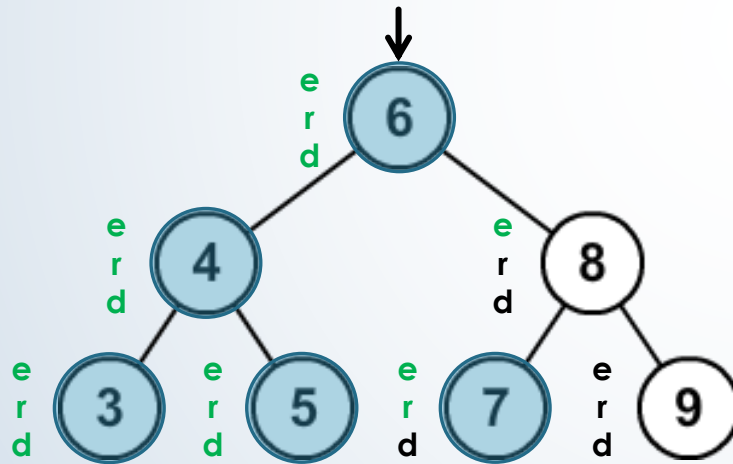


3 4 5 6

Árvore Binária

Percursos – Em ordem (ordem simétrica)

➡ Esquerda ➔ Raiz ➔ Direita

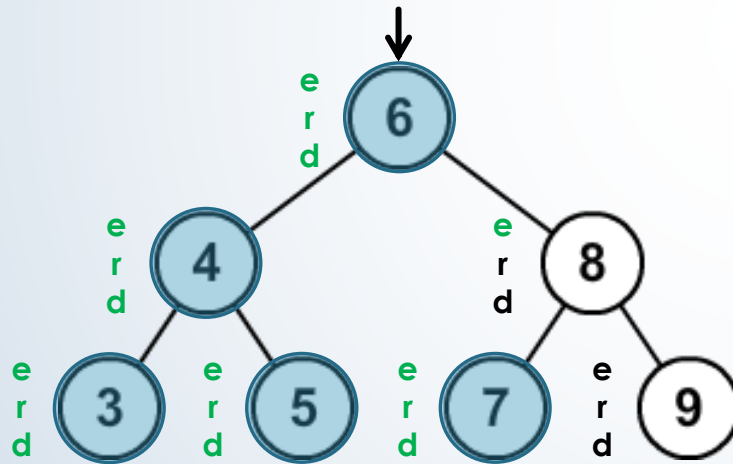


3 4 5 6 7

Árvore Binária

Percursos – Em ordem (ordem simétrica)

➡ Esquerda ➔ Raiz ➔ Direita

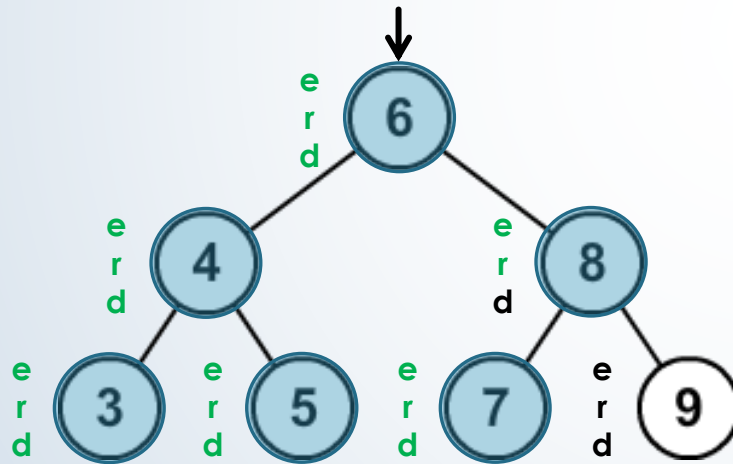


3 4 5 6 7

Árvore Binária

Percursos – Em ordem (ordem simétrica)

➡ Esquerda ➔ Raiz ➔ Direita

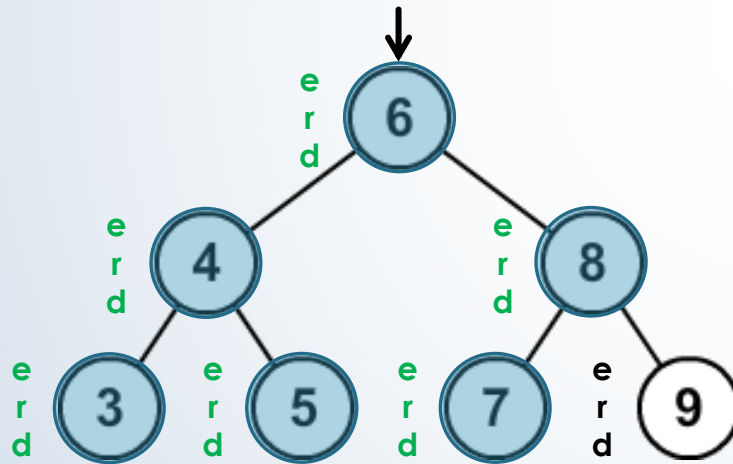


3 4 5 6 7 8

Árvore Binária

Percursos – Em ordem (ordem simétrica)

➡ Esquerda ➔ Raiz ➔ Direita

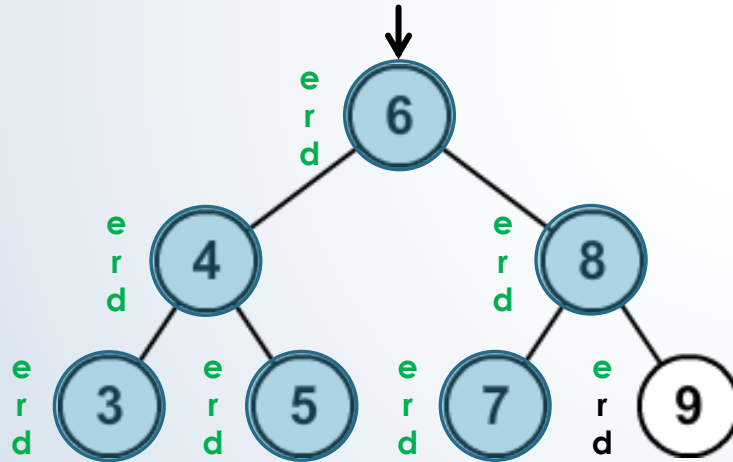


3 4 5 6 7 8

Árvore Binária

Percursos – Em ordem (ordem simétrica)

➡ Esquerda ➔ Raiz ➔ Direita

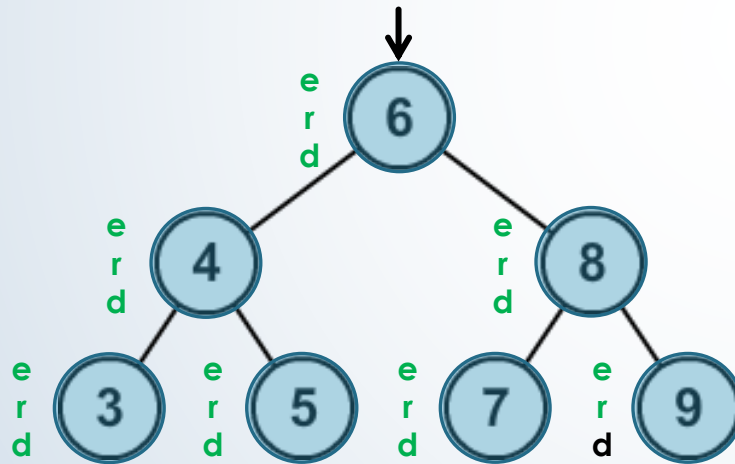


3 4 5 6 7 8

Árvore Binária

Percursos – Em ordem (ordem simétrica)

➡ Esquerda ➔ Raiz ➔ Direita

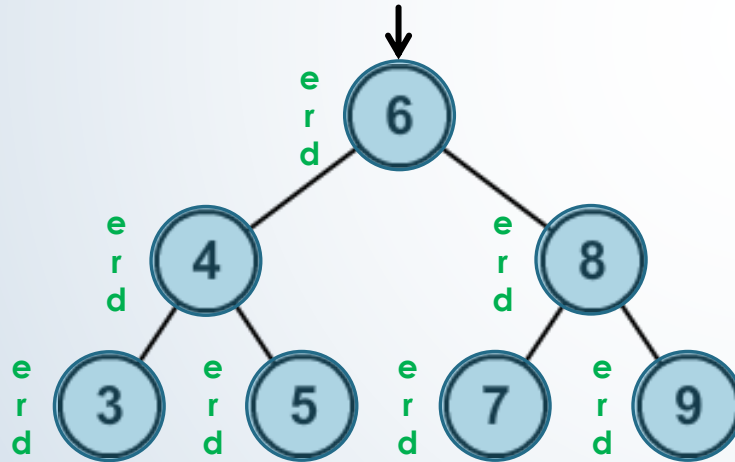


3 4 5 6 7 8 9

Árvore Binária

Percursos – Em ordem (ordem simétrica)

➡ Esquerda ➔ Raiz ➔ Direita

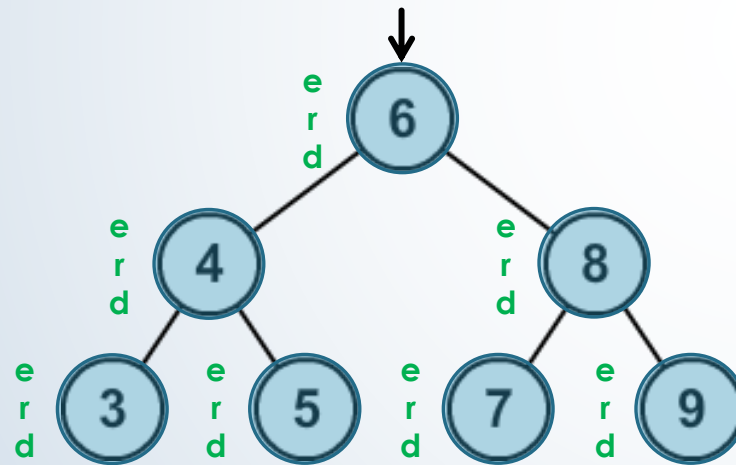


3 4 5 6 7 8 9

Árvore Binária

Percursos – Em ordem (ordem simétrica)

➡ Esquerda ➔ Raiz ➔ Direita

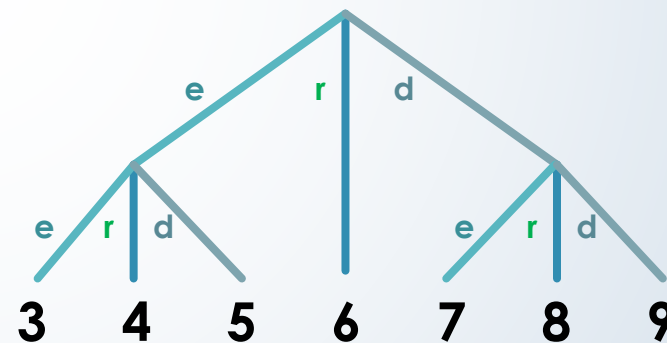
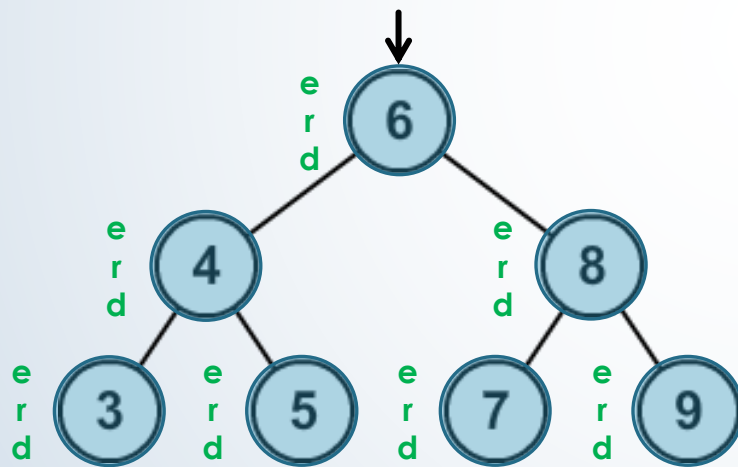


3 4 5 6 7 8 9

Árvore Binária

Percursos – Em ordem (ordem simétrica)

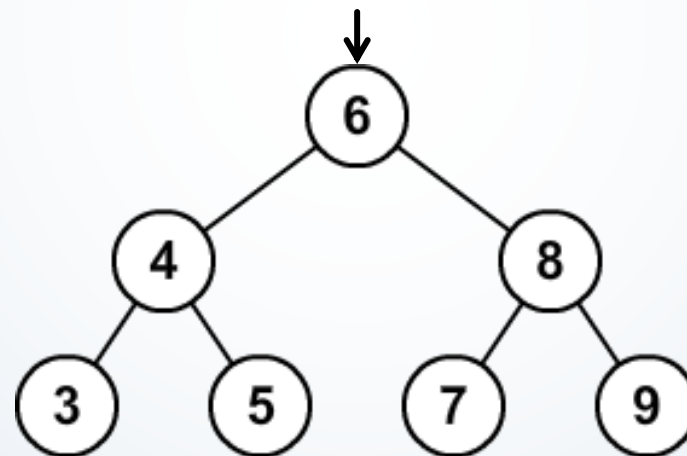
➡ Esquerda ➔ Raiz ➔ Direita



Árvore Binária

Percursos – Pós-ordem (ordem final)

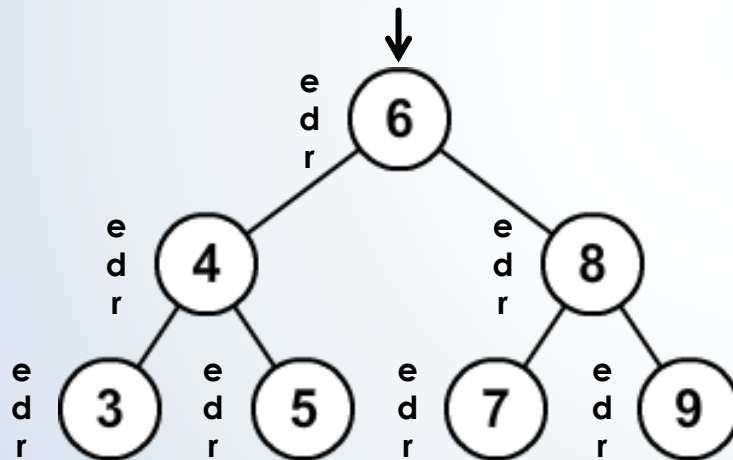
➡ Esquerda ➡ Direita ➡ Raiz



Árvore Binária

Percursos – Pós-ordem (ordem final)

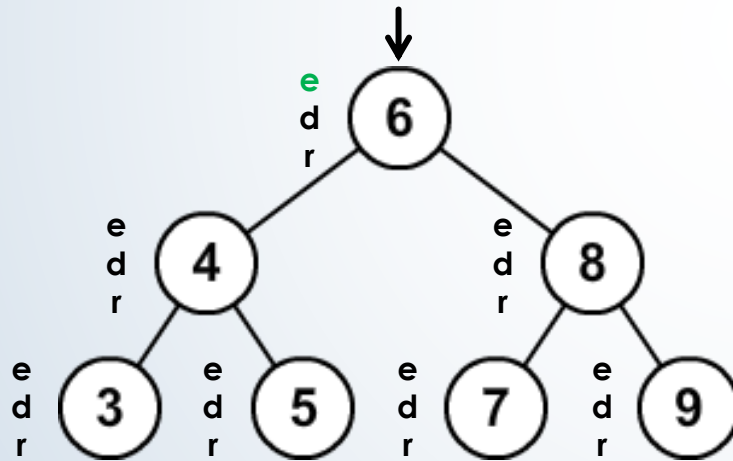
➡ Esquerda ➔ Direita ➔ Raiz



Árvore Binária

Percursos – Pós-ordem (ordem final)

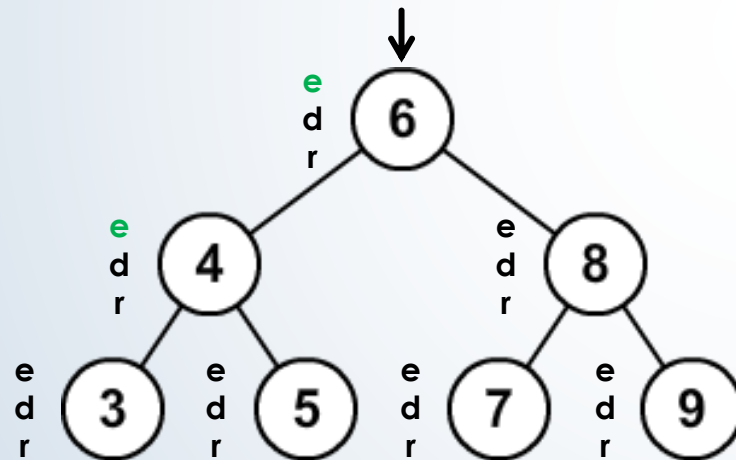
➡ Esquerda ➔ Direita ➔ Raiz



Árvore Binária

Percursos – Pós-ordem (ordem final)

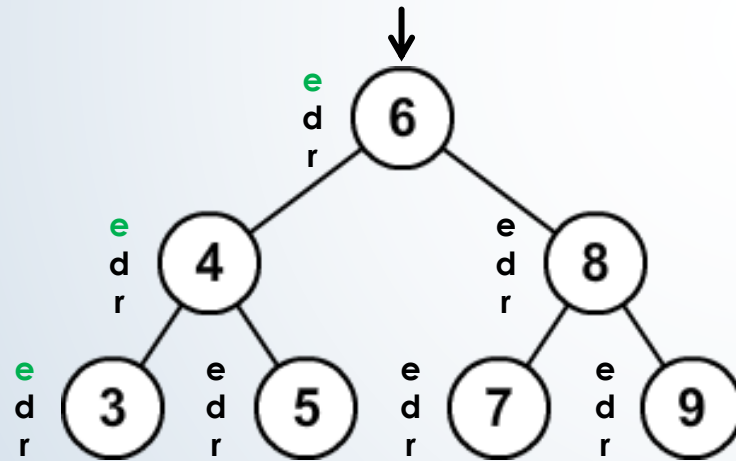
➡ Esquerda ➡ Direita ➡ Raiz



Árvore Binária

Percursos – Pós-ordem (ordem final)

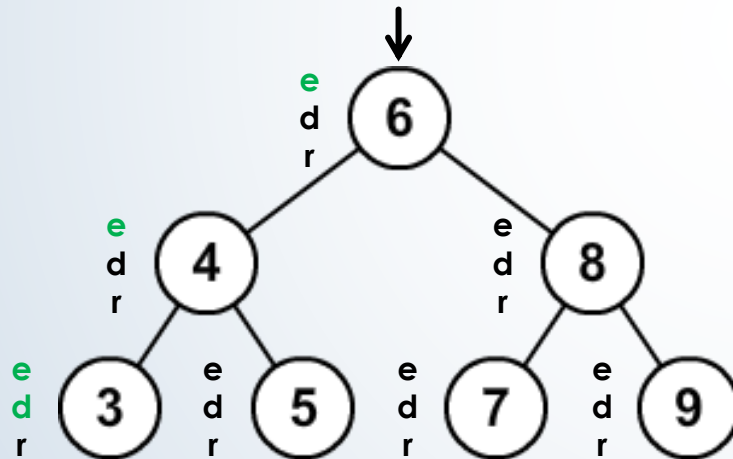
➡ Esquerda ➡ Direita ➡ Raiz



Árvore Binária

Percursos – Pós-ordem (ordem final)

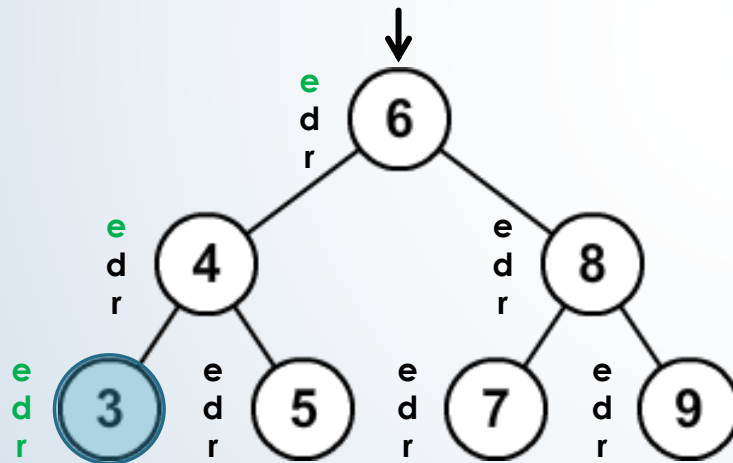
➡ Esquerda ➔ Direita ➔ Raiz



Árvore Binária

Percursos – Pós-ordem (ordem final)

➡ Esquerda ➡ Direita ➡ Raiz

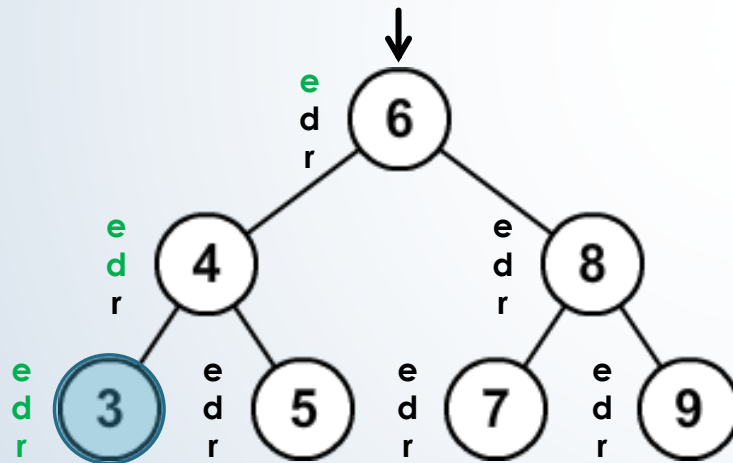


3

Árvore Binária

Percursos – Pós-ordem (ordem final)

➡ Esquerda ➡ Direita ➡ Raiz

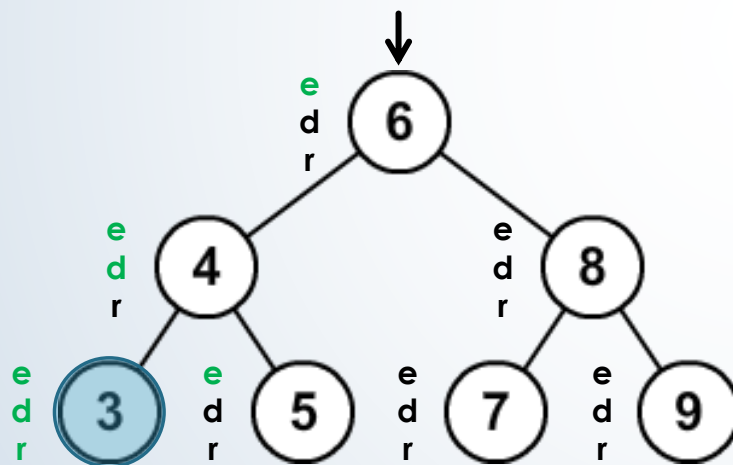


3

Árvore Binária

Percursos – Pós-ordem (ordem final)

➡ Esquerda ➡ Direita ➡ Raiz

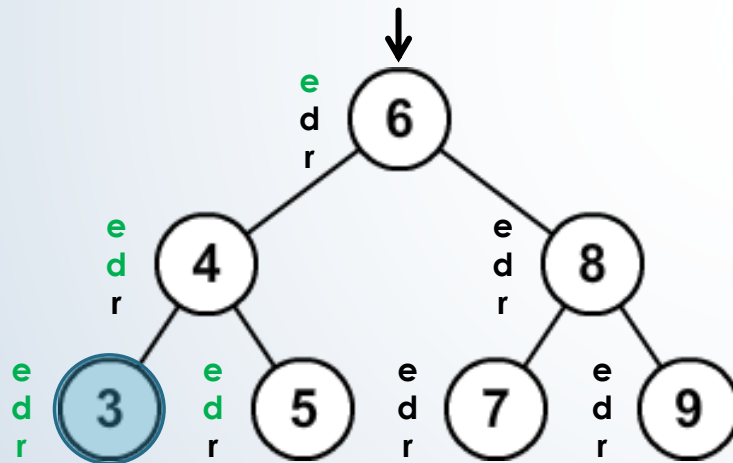


3

Árvore Binária

Percursos – Pós-ordem (ordem final)

➡ Esquerda ➡ Direita ➡ Raiz

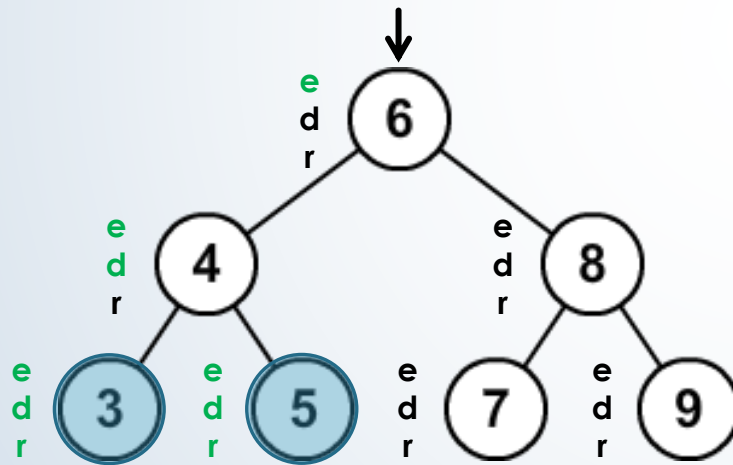


3

Árvore Binária

Percursos – Pós-ordem (ordem final)

➡ Esquerda ➔ Direita ➔ Raiz

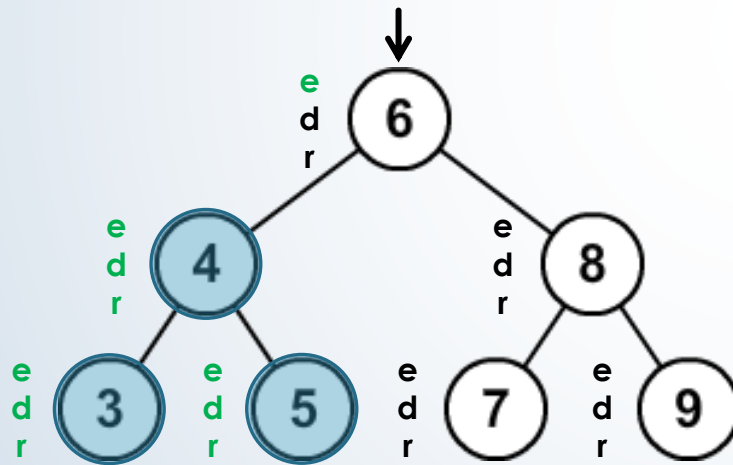


3 5

Árvore Binária

Percursos – Pós-ordem (ordem final)

➡ Esquerda ➔ Direita ➔ Raiz

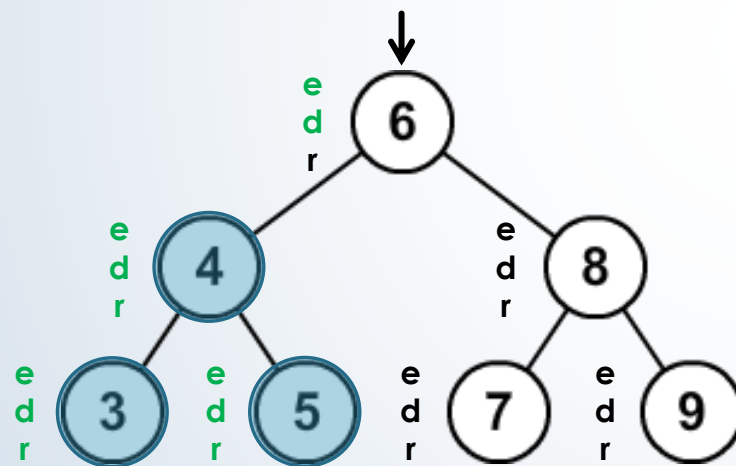


3 5 4

Árvore Binária

Percursos – Pós-ordem (ordem final)

➡ Esquerda ➔ Direita ➔ Raiz

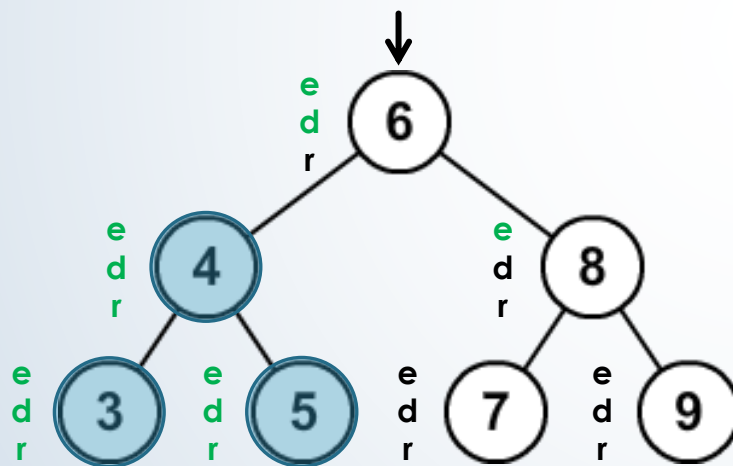


3 5 4

Árvore Binária

Percursos – Pós-ordem (ordem final)

➡ Esquerda ➔ Direita ➔ Raiz

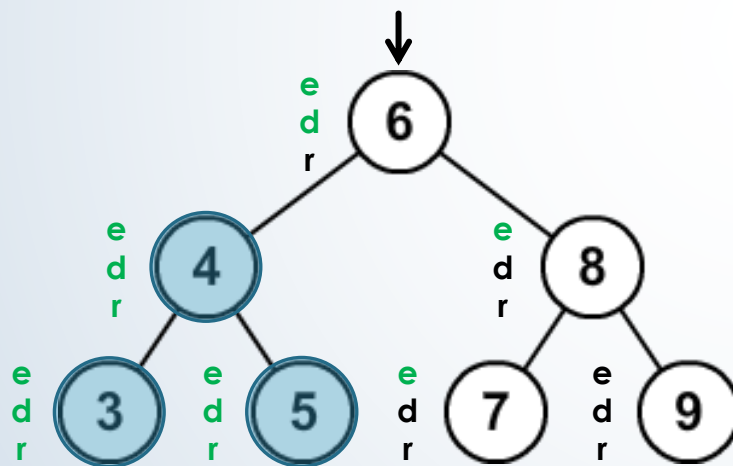


3 5 4

Árvore Binária

Percursos – Pós-ordem (ordem final)

➡ Esquerda ➡ Direita ➡ Raiz

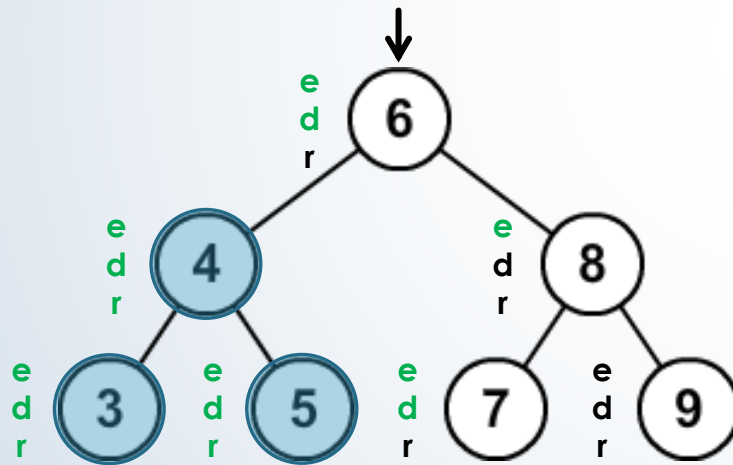


3 5 4

Árvore Binária

Percursos – Pós-ordem (ordem final)

➡ Esquerda ➔ Direita ➔ Raiz

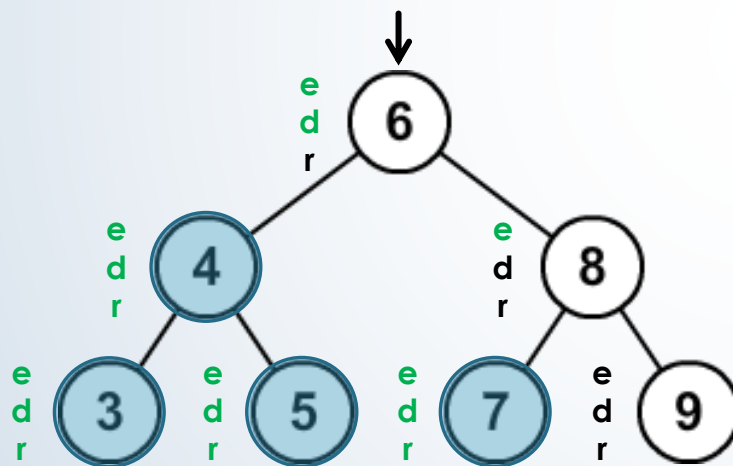


3 5 4

Árvore Binária

Percursos – Pós-ordem (ordem final)

➡ Esquerda ➡ Direita ➡ Raiz

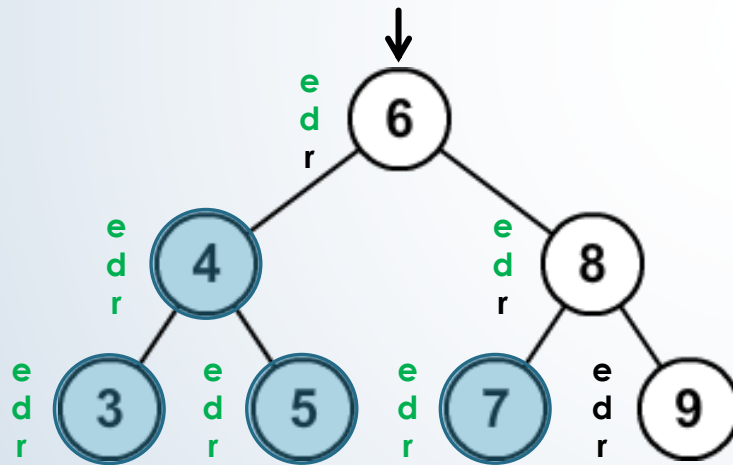


3 5 4 7

Árvore Binária

Percursos – Pós-ordem (ordem final)

➡ Esquerda ➡ Direita ➡ Raiz

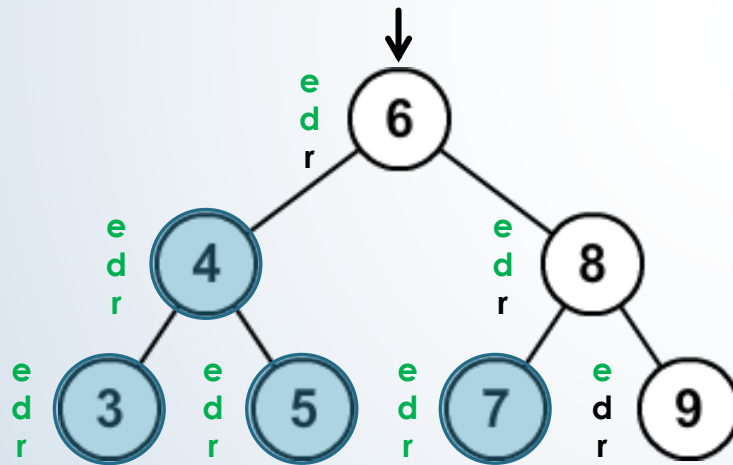


3 5 4 7

Árvore Binária

Percursos – Pós-ordem (ordem final)

➡ Esquerda ➡ Direita ➡ Raiz

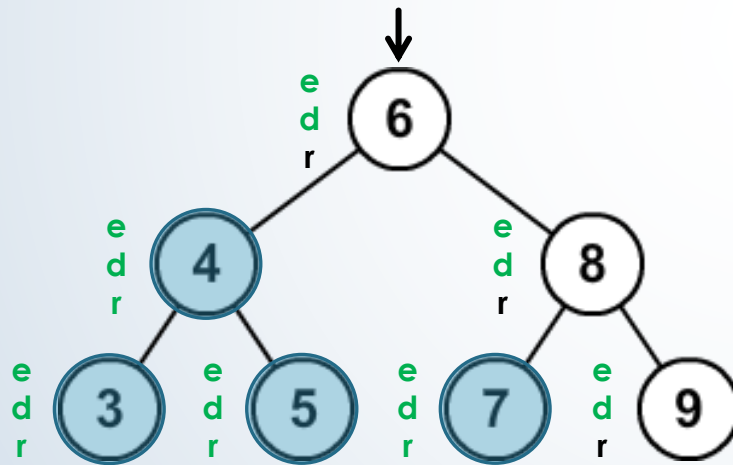


3 5 4 7

Árvore Binária

Percursos – Pós-ordem (ordem final)

➡ Esquerda ➡ Direita ➡ Raiz

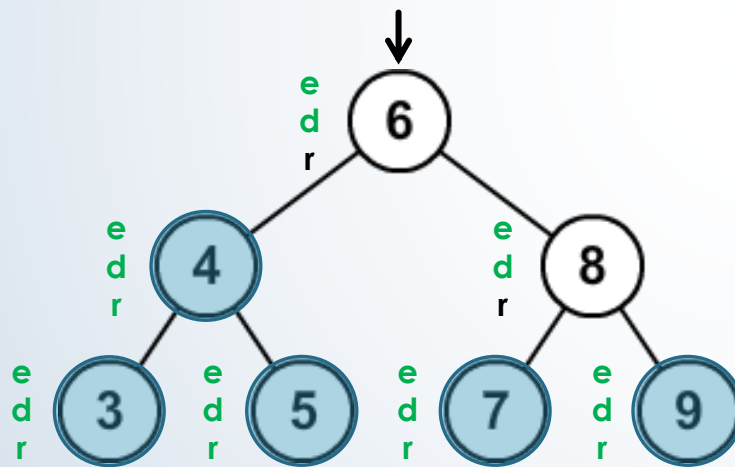


3 5 4 7

Árvore Binária

Percursos – Pós-ordem (ordem final)

➡ Esquerda ➡ Direita ➡ Raiz

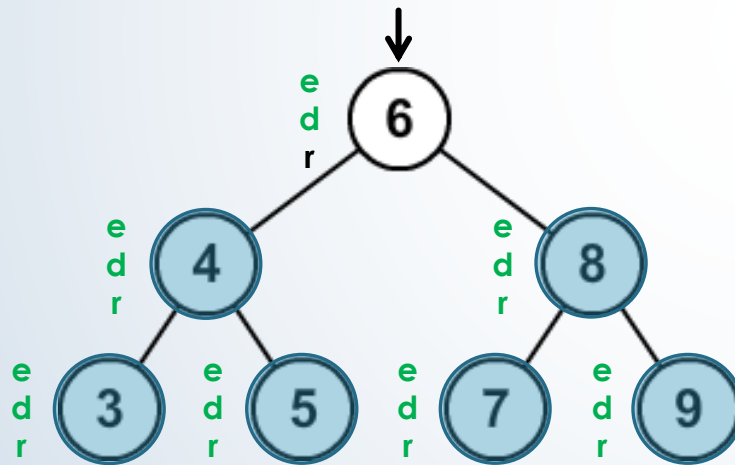


3 5 4 7 9

Árvore Binária

Percursos – Pós-ordem (ordem final)

➡ Esquerda ➡ Direita ➡ Raiz

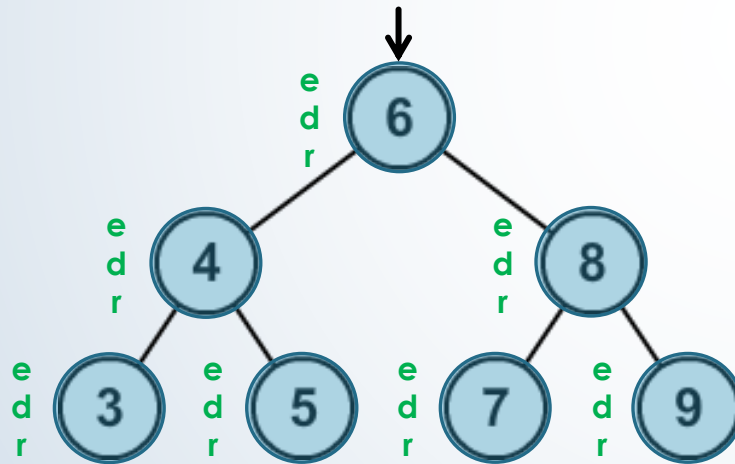


3 5 4 7 9 8

Árvore Binária

Percursos – Pós-ordem (ordem final)

➡ Esquerda ➡ Direita ➡ Raiz

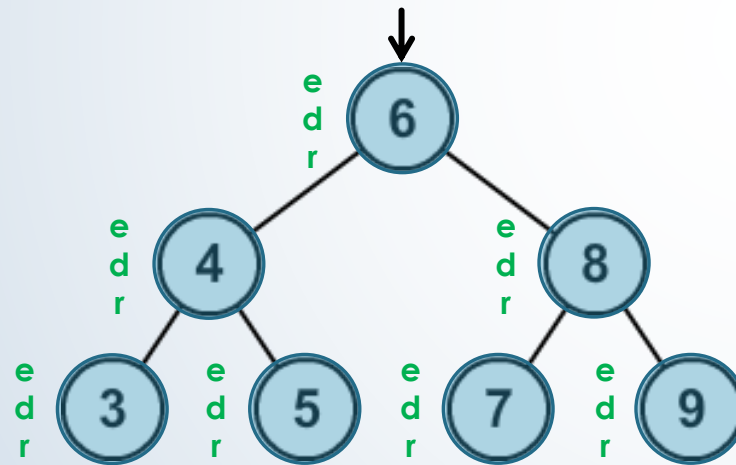


3 5 4 7 9 8 6

Árvore Binária

Percursos – Pós-ordem (ordem final)

➡ Esquerda ➡ Direita ➡ Raiz

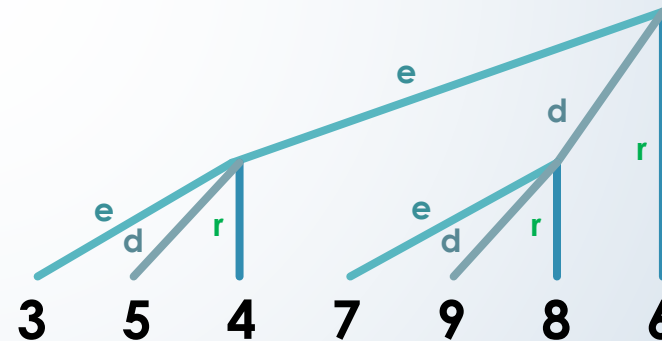
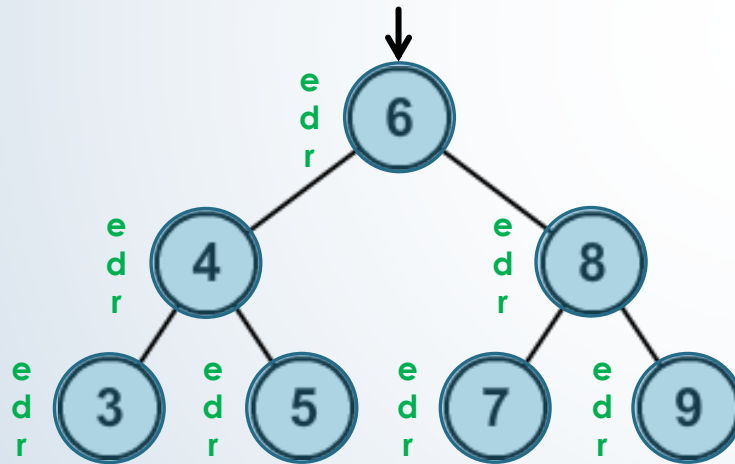


3 5 4 7 9 8 6

Árvore Binária

Percursos – Pós-ordem (ordem final)

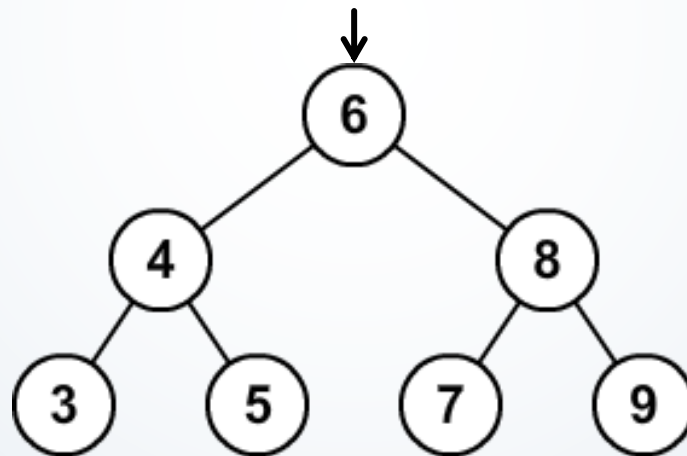
➡ Esquerda ➡ Direita ➡ Raiz



Árvore Binária

Percursos – Em nível

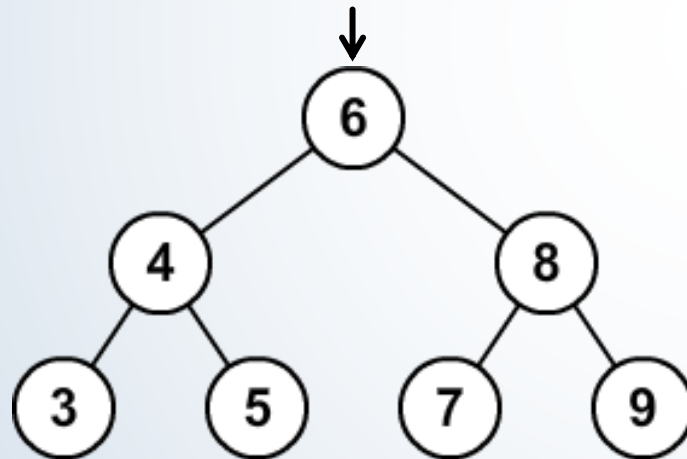
- Nível a nível, iniciando no nível 0, da esquerda para a direita.



Árvore Binária

Percursos – Em nível

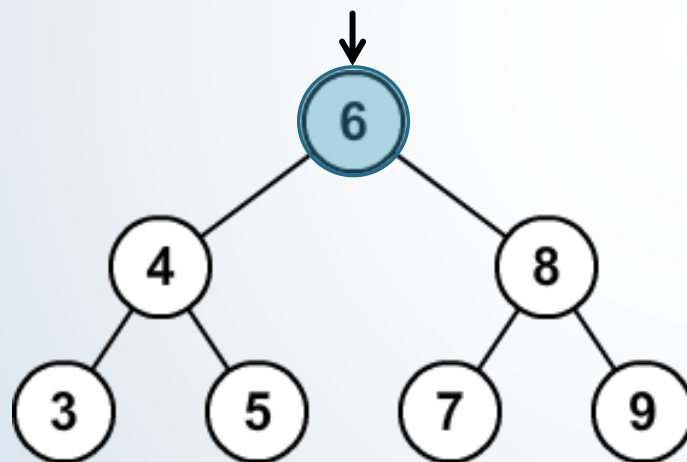
- Nível a nível, iniciando no nível 0, da esquerda para a direita.



Árvore Binária

Percursos – Em nível

- Nível a nível, iniciando no nível 0, da esquerda para a direita.

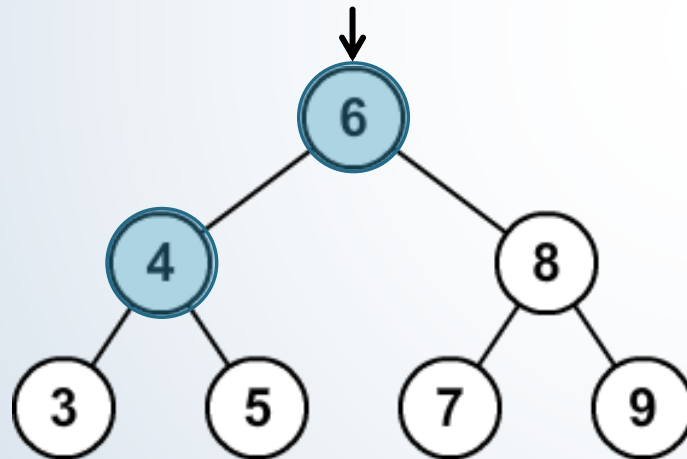


6

Árvore Binária

Percursos – Em nível

- Nível a nível, iniciando no nível 0, da esquerda para a direita.

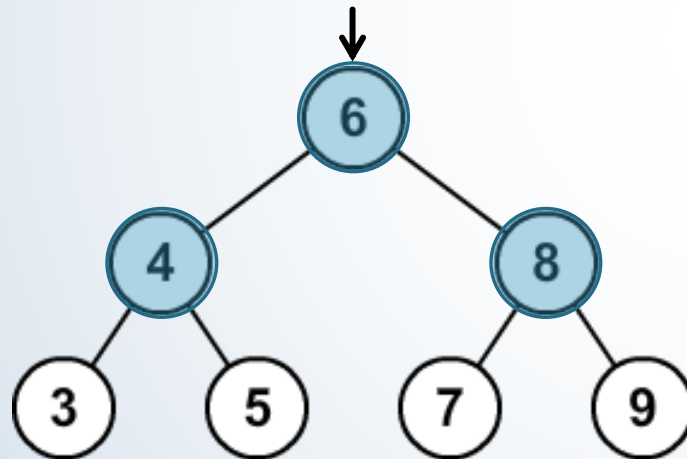


6 4

Árvore Binária

Percursos – Em nível

- Nível a nível, iniciando no nível 0, da esquerda para a direita.

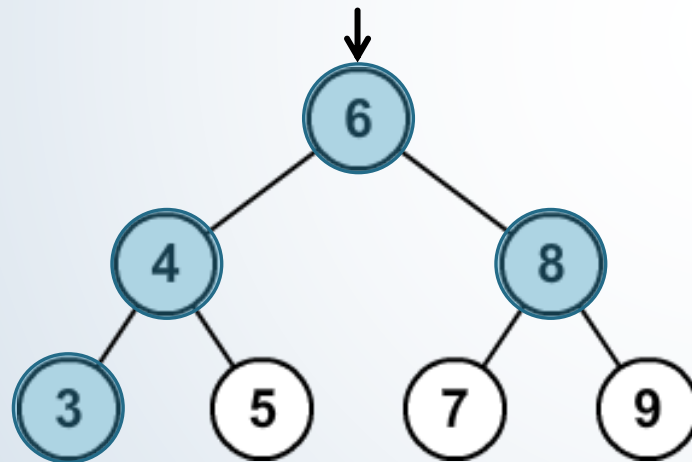


6 4 8

Árvore Binária

Percursos – Em nível

- Nível a nível, iniciando no nível 0, da esquerda para a direita.

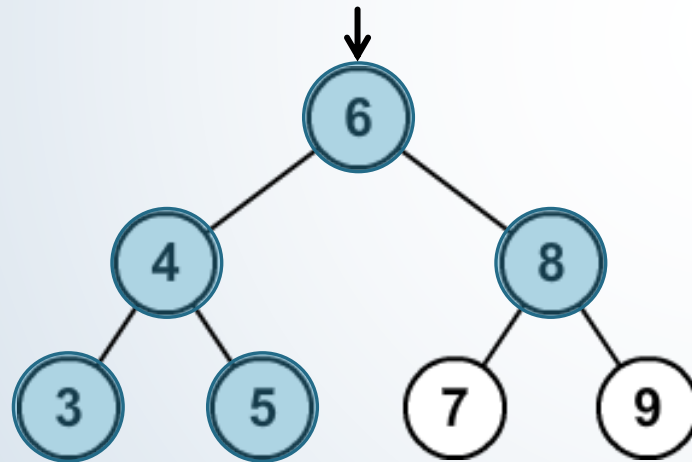


6 4 8 3

Árvore Binária

Percursos – Em nível

- Nível a nível, iniciando no nível 0, da esquerda para a direita.

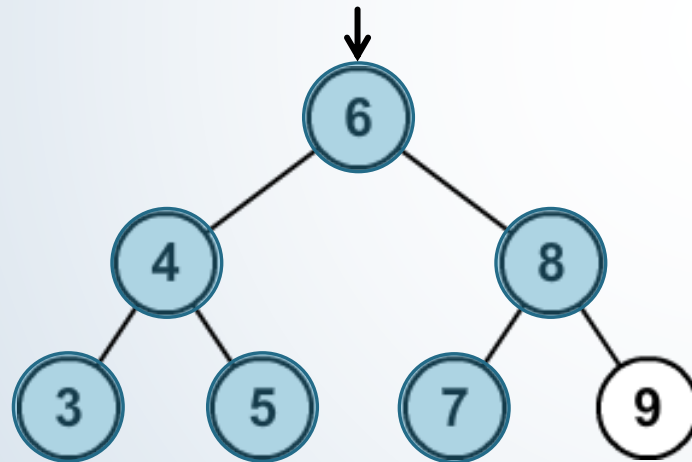


6 4 8 3 5

Árvore Binária

Percursos – Em nível

- Nível a nível, iniciando no nível 0, da esquerda para a direita.

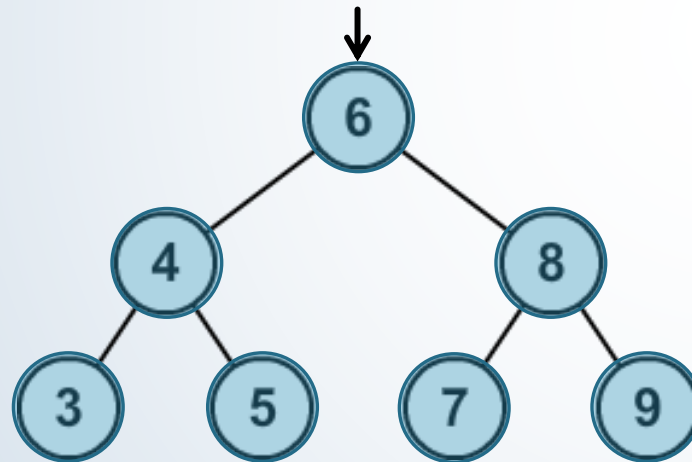


6 4 8 3 5 7

Árvore Binária

Percursos – Em nível

- Nível a nível, iniciando no nível 0, da esquerda para a direita.

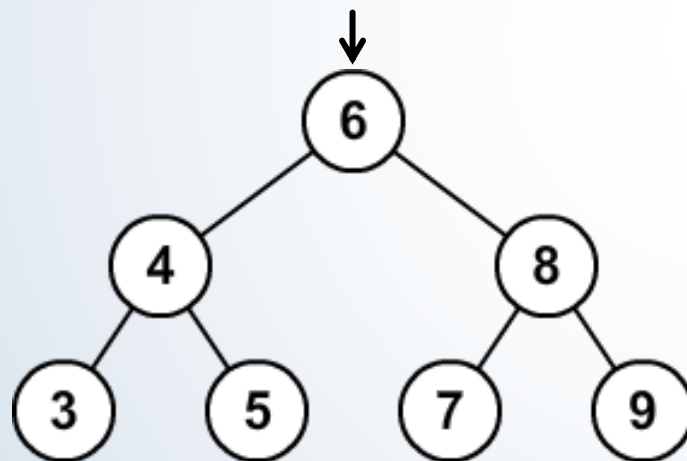


6 4 8 3 5 7 9

Árvore Binária

Percursos – Pré-ordem inverso

➡ Raiz ➡ Direita ➡ Esquerda

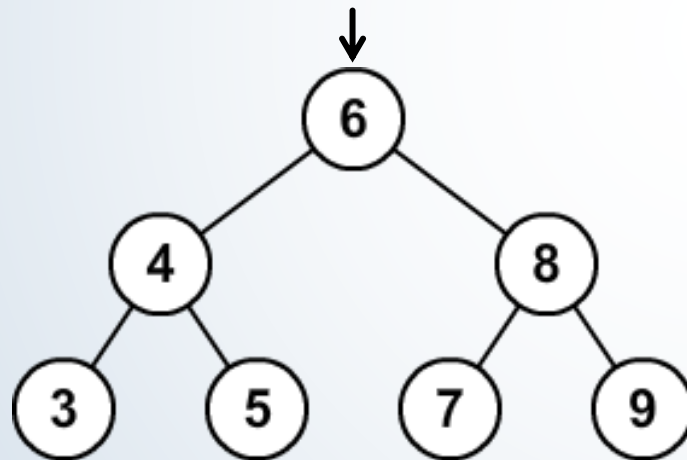


6 8 9 7 4 5 3

Árvore Binária

Percursos – Em ordem (ordem simétrica) inverso

► Direita → Raiz → Esquerda

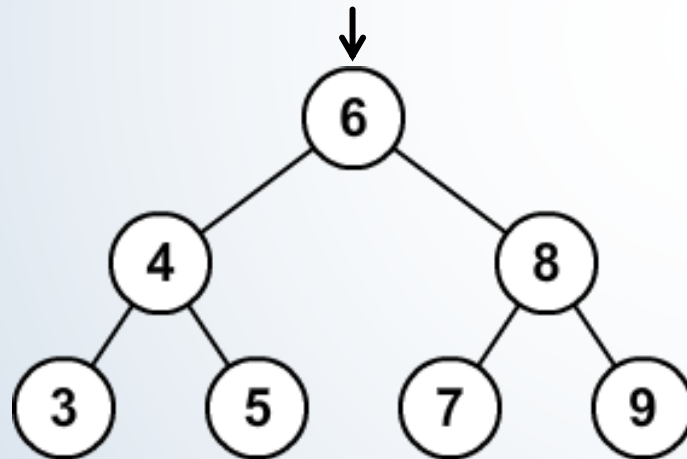


9 8 7 6 5 4 3

Árvore Binária

Percursos – Pós-ordem (ordem final) inverso

► Direita → Esquerda → Raiz

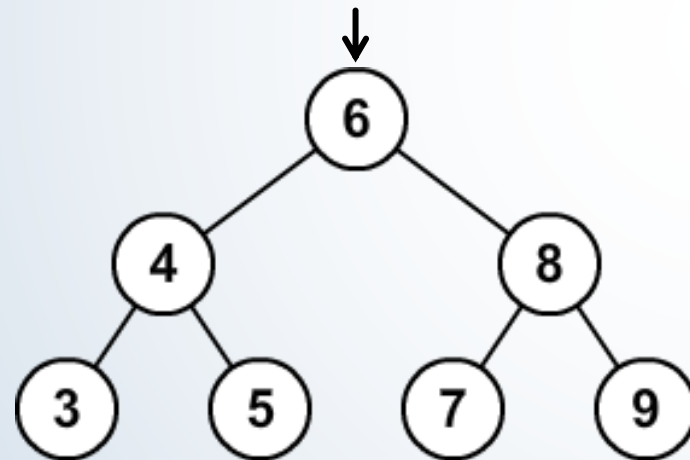


9 7 8 5 3 4 6

Árvore Binária

Percursos – Em nível inverso

- Nível a nível, iniciando no último, da direita para a esquerda.

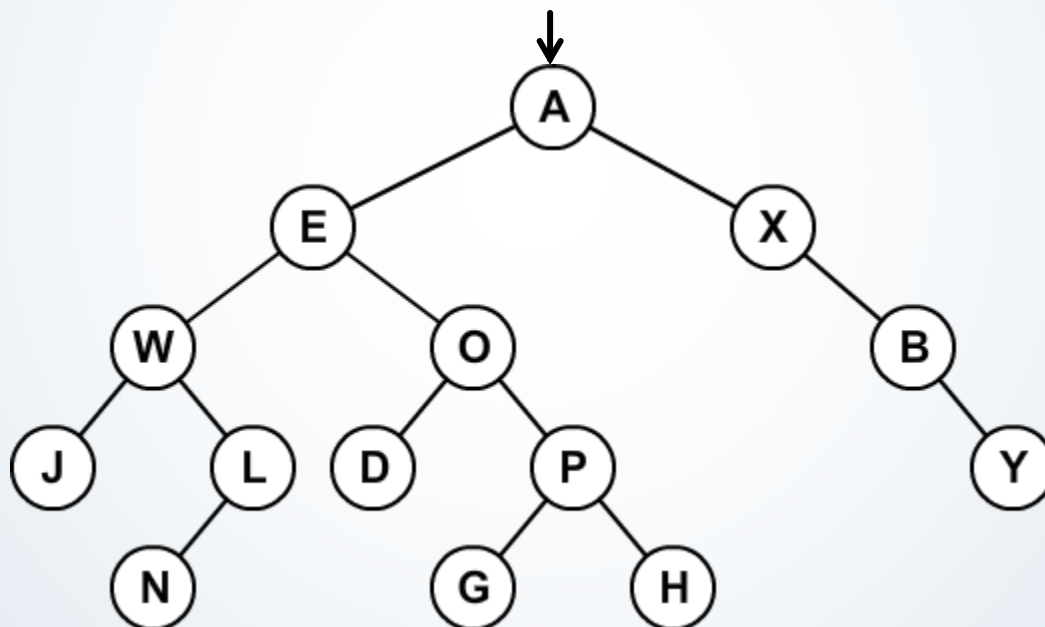


9 7 5 3 8 4 6

Árvore Binária de Busca

Exercícios Escritos

- **Exercício e7.1:** Considere a árvore binária abaixo, em que as chaves são caracteres e estão representadas dentro de cada nó, e responda às perguntas a seguir:



Árvore Binária de Busca

Exercícios Escritos

- a) Quais são os nós da subárvore direita do nó A?
- b) Quais são os nós da subárvore direita do nó O?
- c) Quais são os nós da subárvore esquerda do nó E?
- d) Qual é o grau do nó E?
- e) Qual é o grau do nó L?
- f) Qual é o grau do nó N?
- g) Qual é o grau do nó B?
- h) Qual é o grau da árvore?
- i) Para cada nó da árvore, diga: qual é o seu pai, qual(is) são seu(s) filho(s) e se o nó é um nó folha (nó terminal).
- j) Quais são os nós ancestrais do nó P?

Árvore Binária de Busca

Exercícios Escritos

- k) Quais são os nós ancestrais do nó G?
- l) Quais são os nós ancestrais do nó N?
- m) Quais são os nós ancestrais do nó Y?
- n) Quais são os nós ancestrais do nó A?
- o) Quais são os nós descendentes do nó Y?
- p) Quais são os nós descendentes do nó O?
- q) Quais são os nós descendentes do nó G?
- r) Quais são os nós descendentes do nó E?
- s) Quais são os nós descendentes direito do nó O?
- t) Quais são os nós descendentes direito do nó L?

Árvore Binária de Busca

Exercícios Escritos

- u) Quais são os nós descendentes direito do nó W?
- v) Quais são os nós descendentes esquerdo do nó E?
- w) Quais são os nós descendentes esquerdo do nó O?
- x) Quais são os nós descendentes esquerdo do nó L?
- y) Qual a altura da árvore?
- z) Qual o nível do nó Y?
- aa) Qual o nível do nó P?
- bb) Qual o nível do nó W?
- cc) Qual o nível do nó J?
- dd) Qual o nível do nó A?

Árvore Binária de Busca

Exercícios Escritos

- **Exercício e7.2:** Para cada série de chaves, apresentados a seguir, que serão inseridas em uma árvore binária de busca, desenhe a árvore correspondente.
- a) 4, 8, 7, 6, 10, 9, 3, 2.
 - b) 5, 4, 3, 2, 9, 7, 8, 12, -4, 22.
 - c) K, L, X, M, N, O, R, Q, T, A.
 - d) R, G, T, A, Q, E, F, O, P, L.
 - e) maria, joão, josé marcos, adriana, murilo.
 - f) urso, cachorro, abelha, cavalo, gato, formiga, leão, galinha, girafa, macaco.

Árvore Binária de Busca

Exercícios Escritos

- **Exercício e7.3:** Para cada árvore do exercício anterior, apresente os 8 percursos estudados, ou seja, pré-ordem, em ordem, pós-ordem, em nível e suas variações inversas.

Árvore Binária de Busca

Exercícios de Implementação

- **Exercício i7.1:** No projeto **ESDC4Aula07**, é fornecido o esqueleto de uma classe chamada **BinarySearchTreeSort**, que contém o método

```
public static <Type extends Comparable<Type>> void sort( Type[] array )
```

- Sua tarefa é implementar um algoritmo de ordenação que use uma árvore binária de busca para ordenar o array passado como parâmetro. Os testes de unidade para a aceitação ou não do que deve ser feito foram implementados. Use a implementação **BinarySearchTreeDupKeys** contida no projeto **ESDC4Aula07**. Essa implementação é uma versão simplificada da classe **BinarySearchTree** contida no projeto de estruturas de dados disponível no GitHub, com a diferença que aceita chaves duplicadas, característica necessária para realizar a ordenação.

Bibliografia

SEDGEWICK, R.; WAYNE, K. **Algorithms**. 4. ed. Boston: Pearson Education, 2011. 955 p.

GOODRICH M. T.; TAMASSIA, R. **Estruturas de Dados & Algoritmos em Java**. Porto Alegre: Bookman, 2013. 700 p.

CORMEN, T. H.; LEISERSON, C. E.; RIVEST, R. L.; STEIN, C. **Algoritmos – Teoria e Prática**. 3. ed. São Paulo: GEN LTC, 2012. 1292 p.