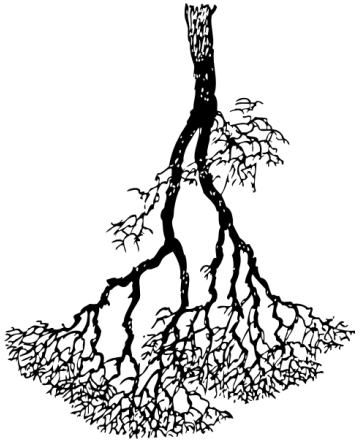
Árvores Binárias de Busca (BST)



Árvores de busca

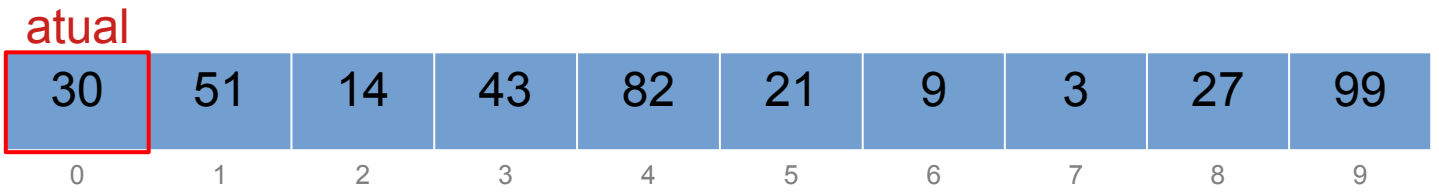
- As árvores binárias de busca são projetadas para um acesso rápido à informação.
  - Binary Search Tree (BST)
- Uma BST é vista como um conjunto de registros, onde cada registro é denominado de *nó* e satisfaz certas condições
  - Cada nó X possui dois elementos filhos
    - O elemento à sua esquerda, considerado o filho esquerdo, é menor que X
    - O elemento à sua direita, considerado o filho direito, é maior ou igual a X

Árvores de busca

- Uma estrutura para uma árvore binária geralmente é simples e apresenta apenas 3 campos:

- O campo ***int value*** apenas ilustra um tipo de dados simples
  - Você pode substituí-lo por qualquer tipo que julgar necessário: char, double, struct…
- Para todo nó de uma árvore de busca binária:
  - O elemento à esquerda deve ser menor que o pai
  - O elemento à direita deve ser maior ou igual ao pai

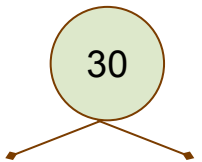
Árvores de busca (Ex)



Árvores de busca (Ex)

atual

30	51	14	43	82	21	9	3	27	99
0	1	2	3	4	5	6	7	8	9

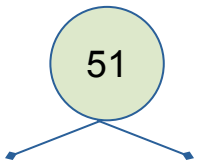


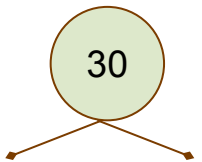
Raiz

Árvores de busca (Ex)

atual

30	51	14	43	82	21	9	3	27	99
0	1	2	3	4	5	6	7	8	9



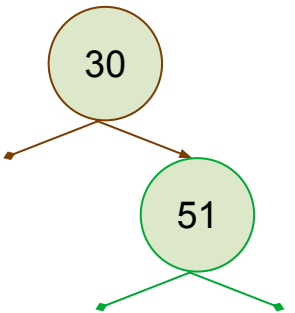


Raiz

Árvores de busca (Ex)

atual

30	51	14	43	82	21	9	3	27	99
0	1	2	3	4	5	6	7	8	9

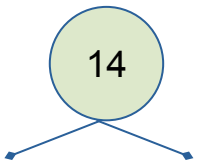


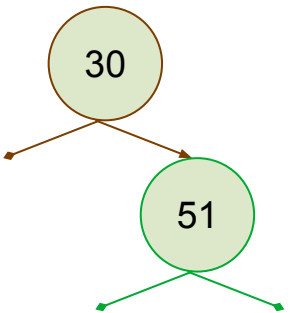
Raiz

Árvores de busca (Ex)

atual

30	51	14	43	82	21	9	3	27	99
0	1	2	3	4	5	6	7	8	9



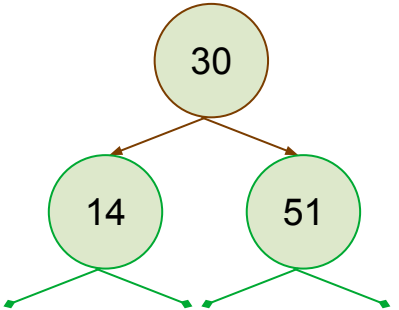


Raiz

Árvores de busca (Ex)

atual

30	51	14	43	82	21	9	3	27	99
0	1	2	3	4	5	6	7	8	9

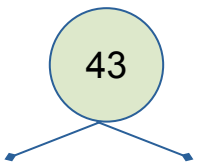


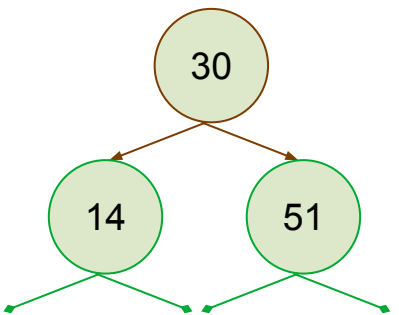
Raiz

Árvores de busca (Ex)

atual

30	51	14	43	82	21	9	3	27	99
0	1	2	3	4	5	6	7	8	9



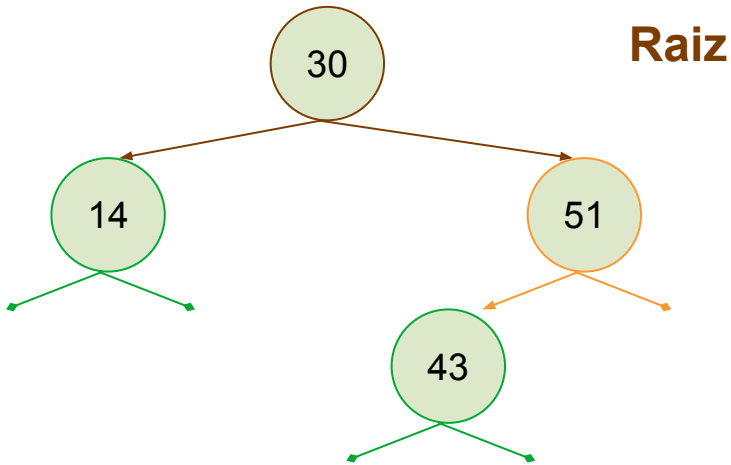


Raiz

Árvores de busca (Ex)

atual

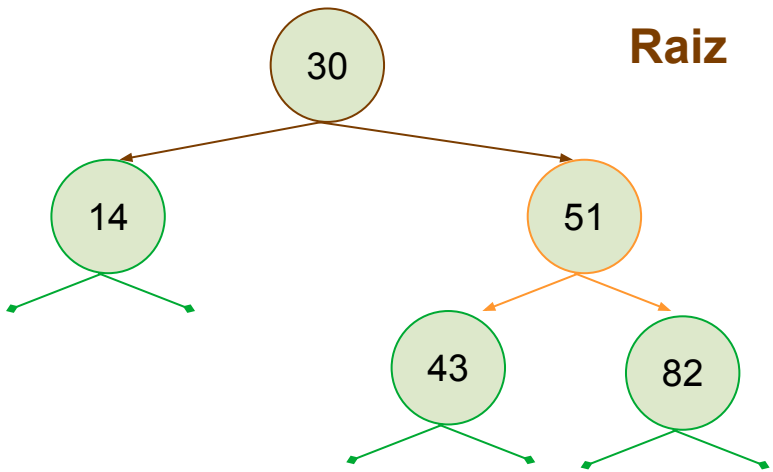
30	51	14	43	82	21	9	3	27	99
0	1	2	3	4	5	6	7	8	9



Árvores de busca (Ex)

atual

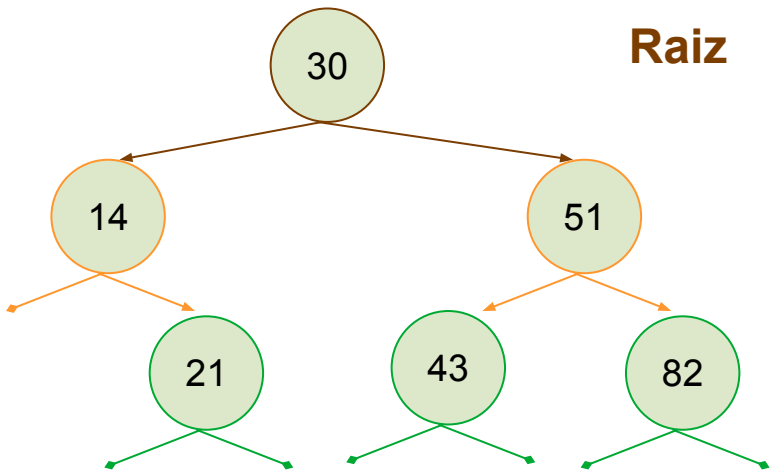
30	51	14	43	82	21	9	3	27	99
0	1	2	3	4	5	6	7	8	9



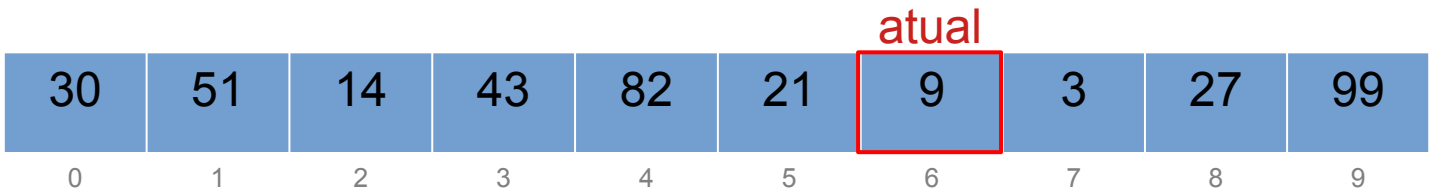
Árvores de busca (Ex)

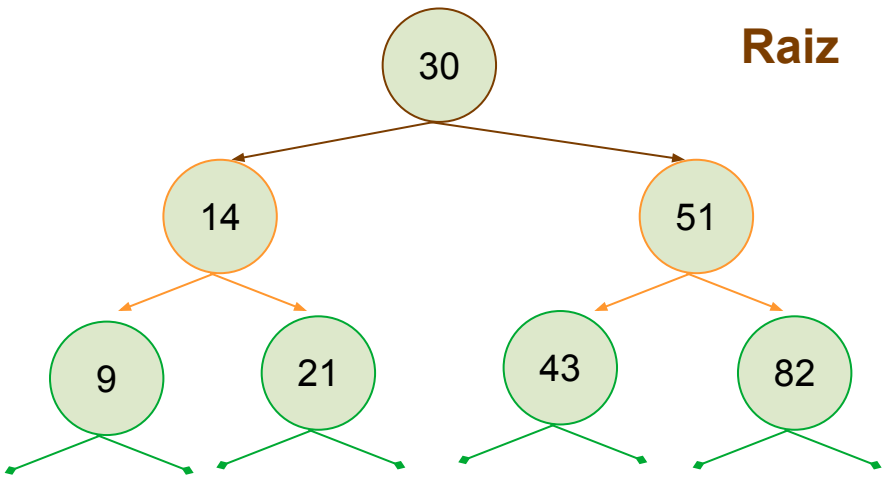
atual

30	51	14	43	82	21	9	3	27	99
0	1	2	3	4	5	6	7	8	9

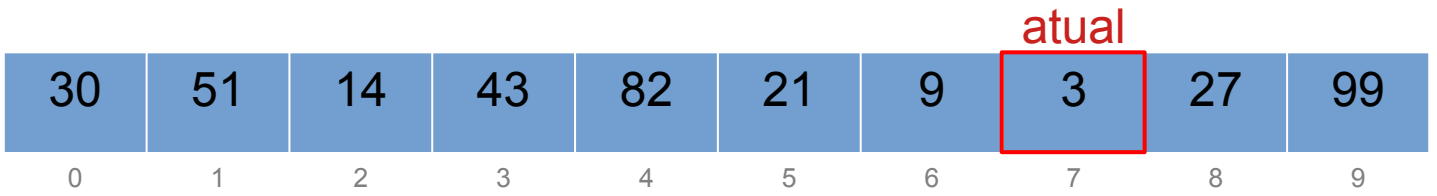


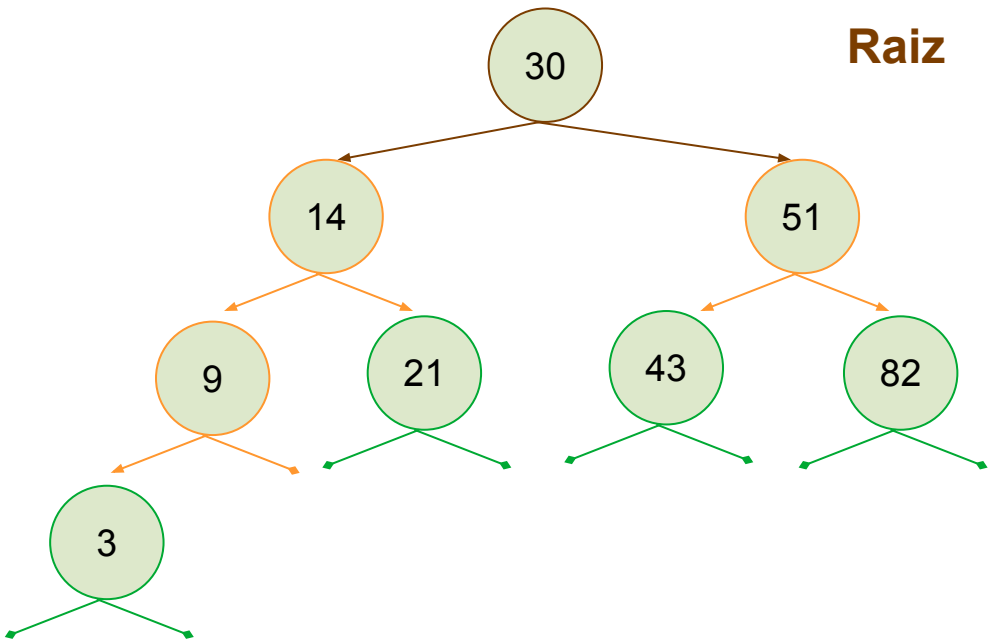
Árvores de busca (Ex)



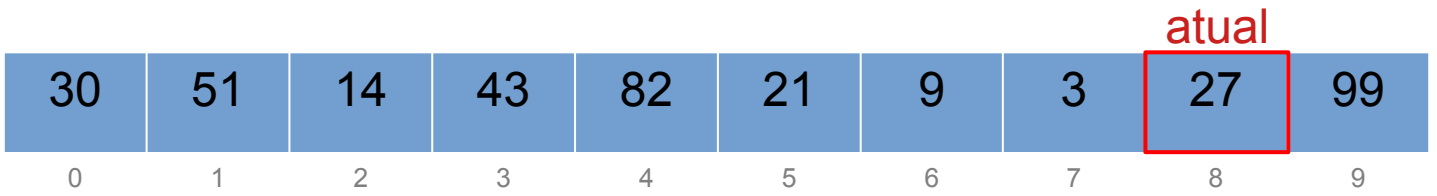


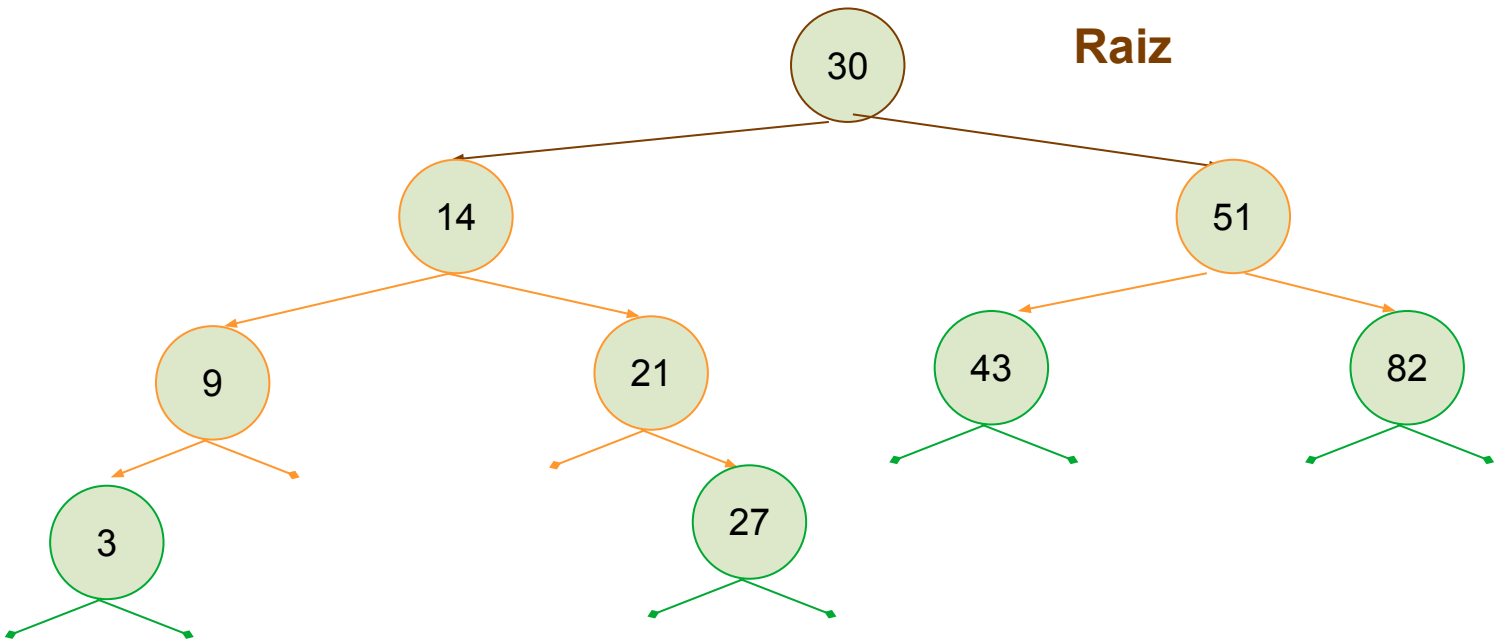
Árvores de busca (Ex)



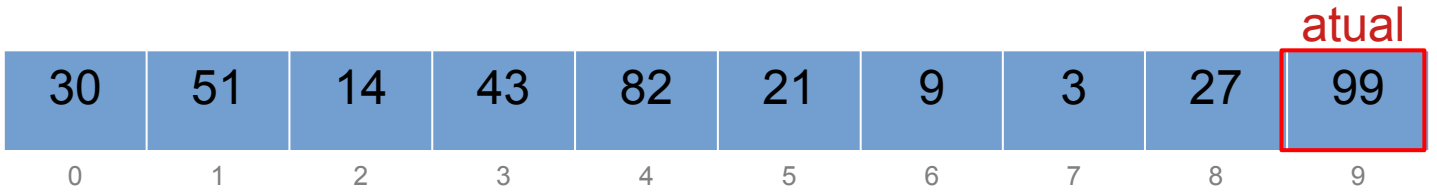


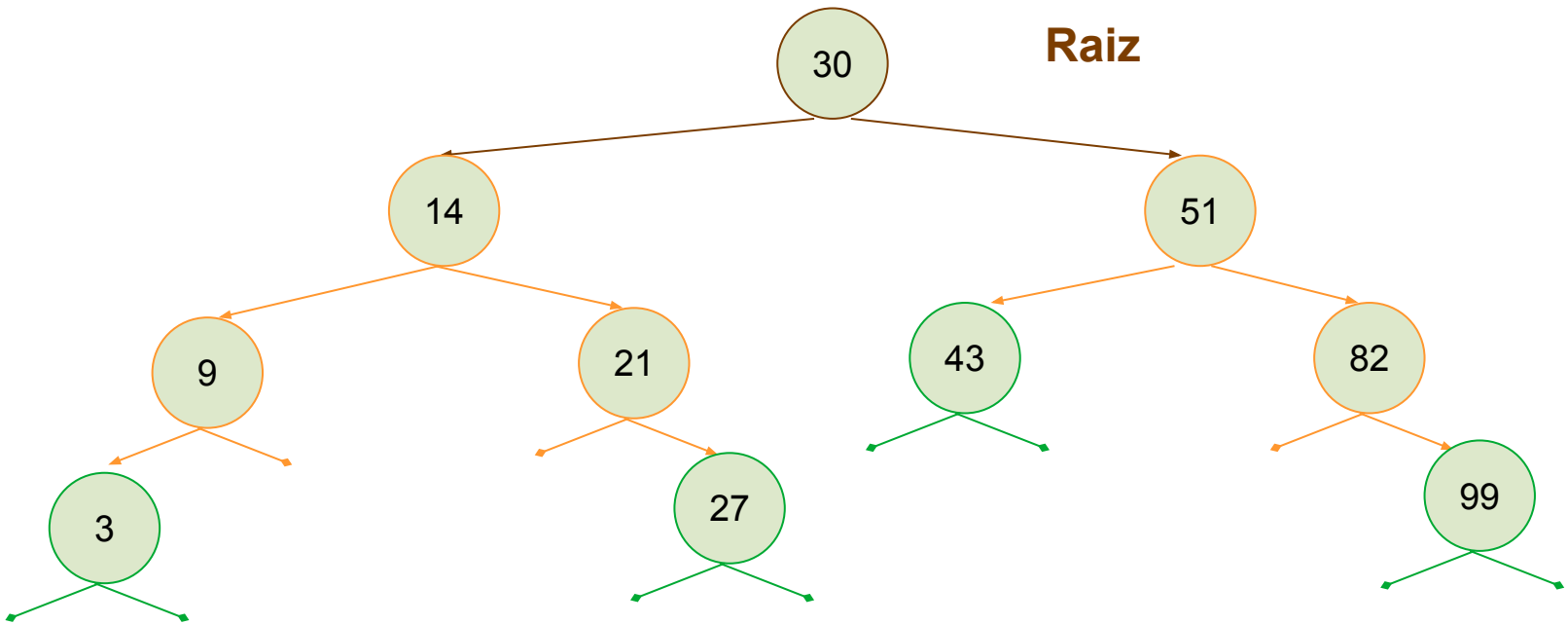
Árvores de busca (Ex)





Árvores de busca (Ex)



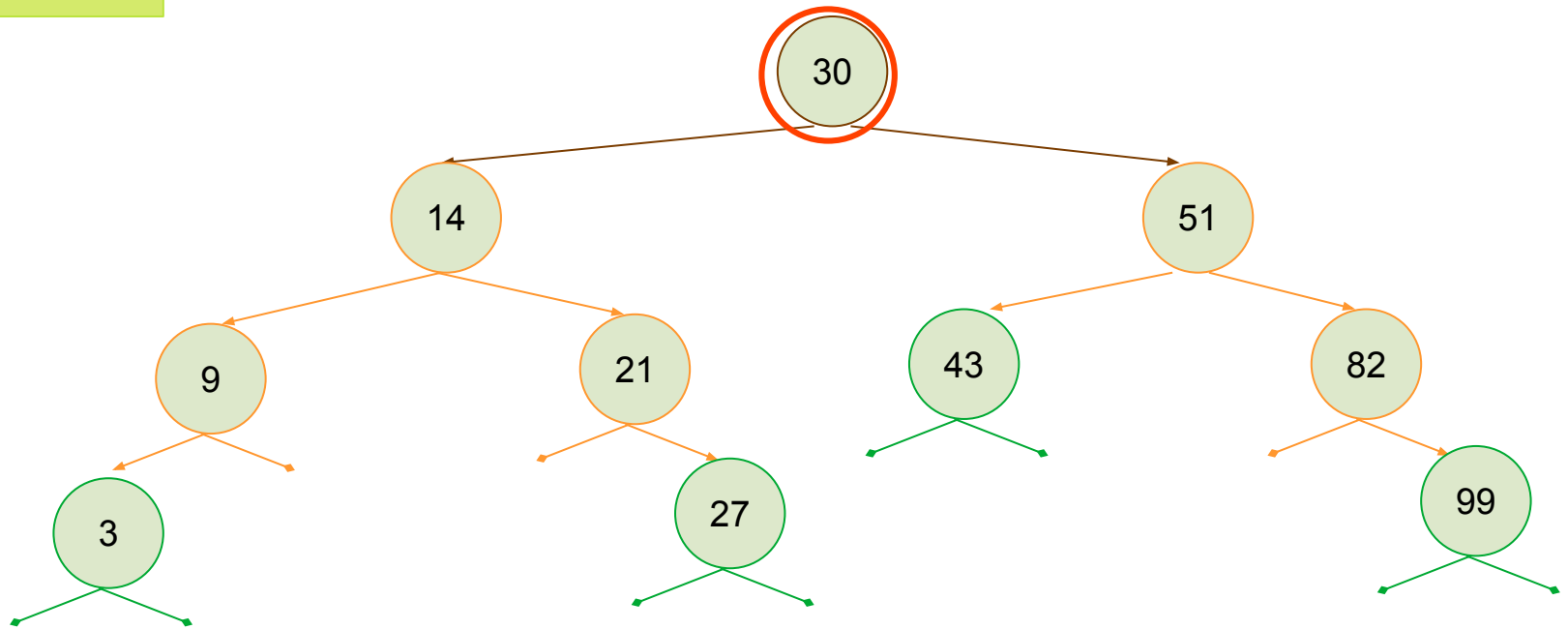


Árvores de busca

```
algoritmo adicionaNo(Node *raiz, Node *new)
inicio
    if (raiz == NULL)
       return new;
    else
       if (raiz→valor >= new→valor)
         raiz→left = adicionaNo(raiz→left, new);
       else
         raiz→right = adicionaNo(raiz→right, new);
       fim se
    fim se
    return raiz;
fimAlgoritmo
```

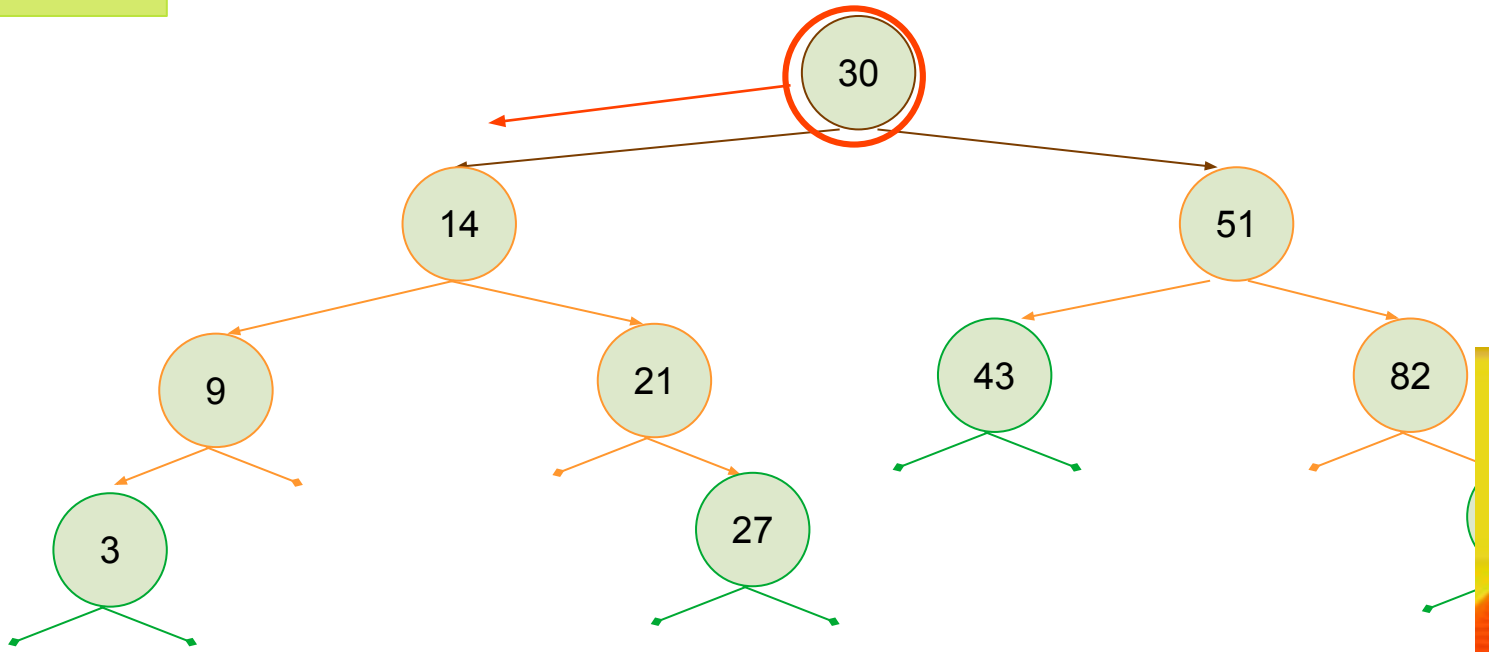
Árvores de busca (Busca)

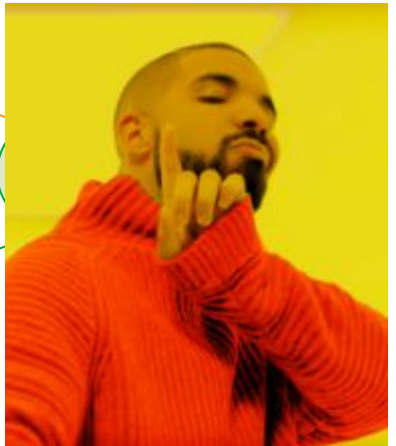
K = 21



Árvores de busca (Busca)

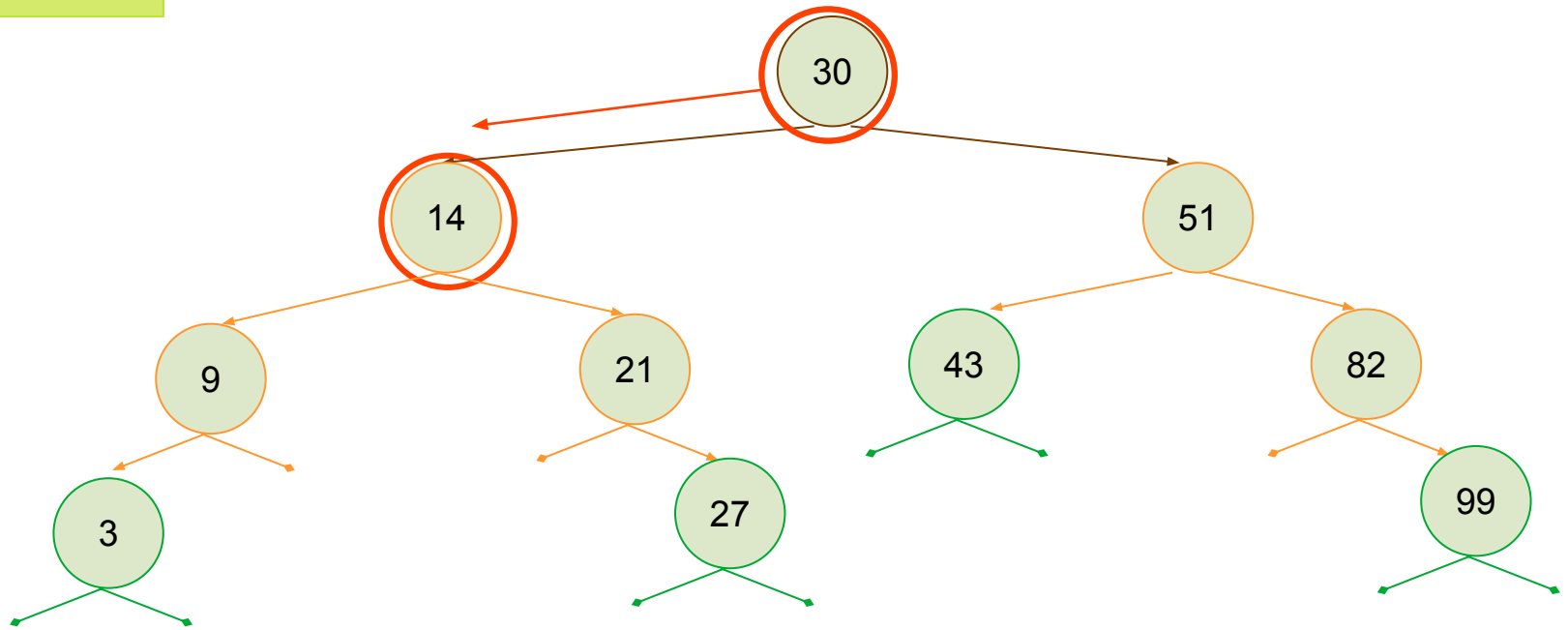
K = 21





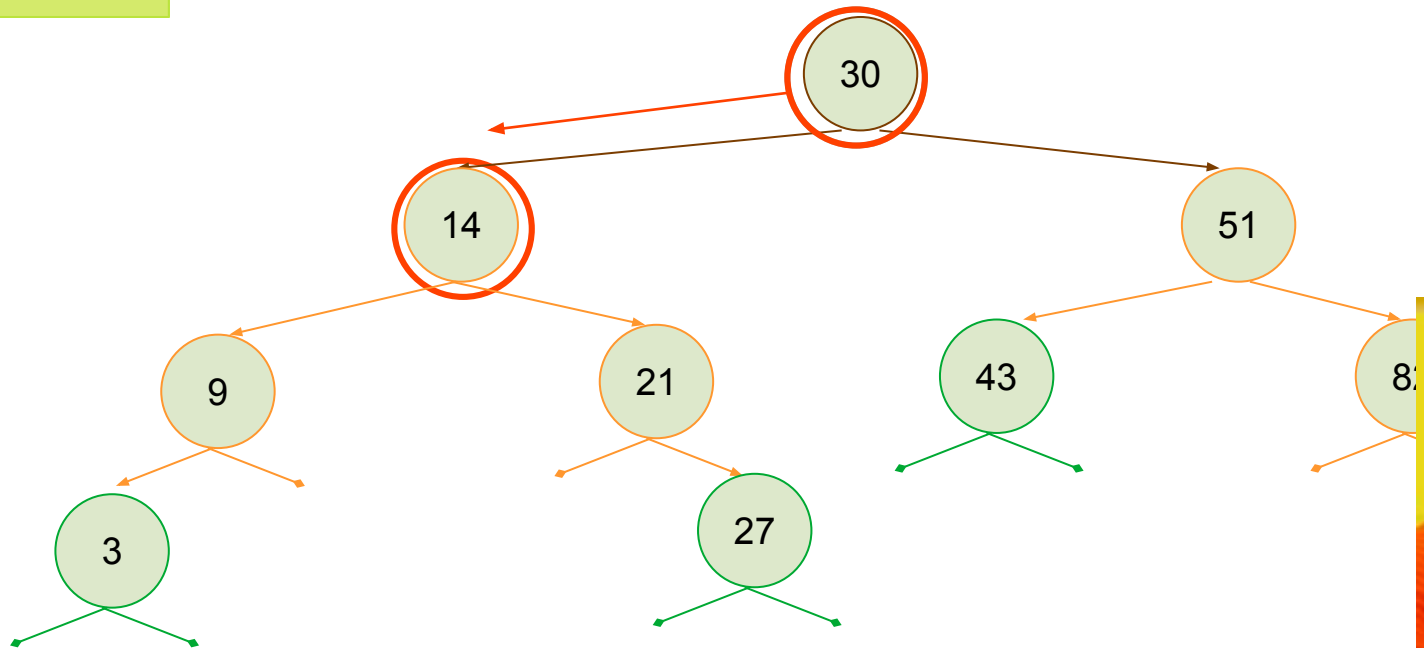
Árvores de busca (Busca)

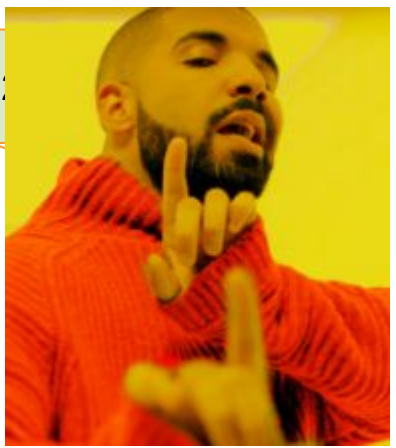
K = 21



Árvores de busca (Busca)

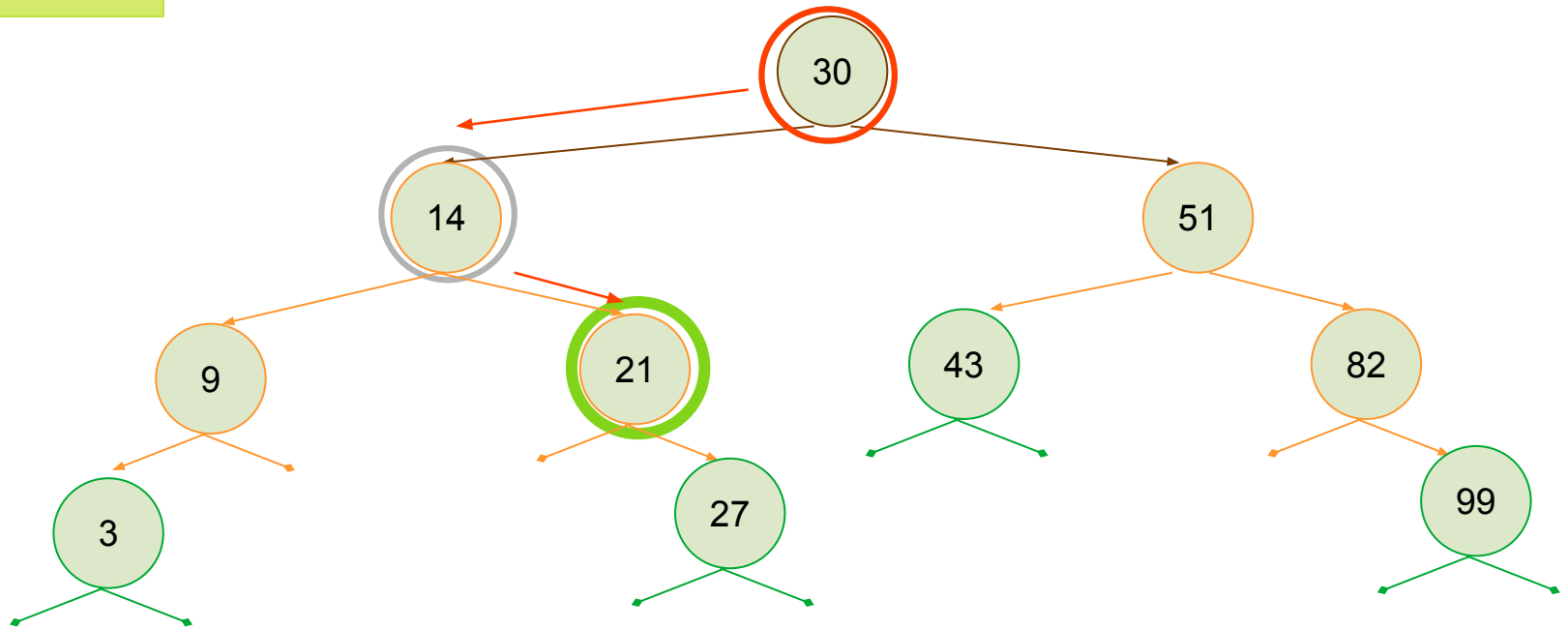
K = 21





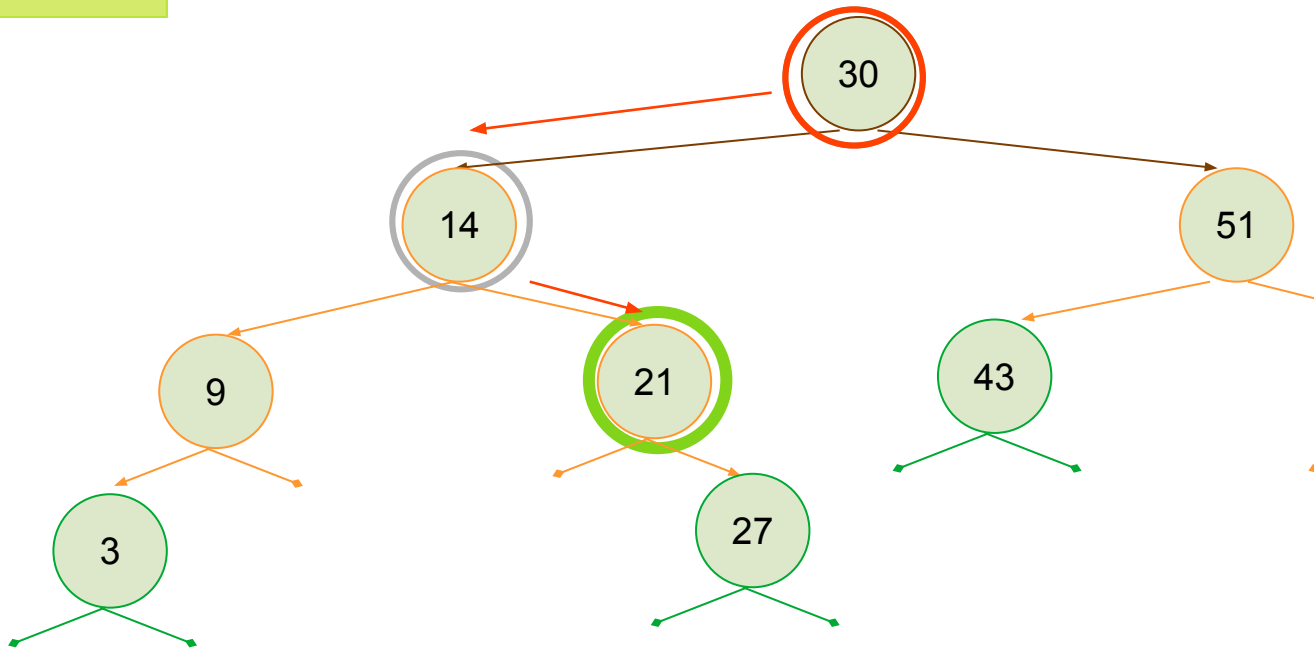
Árvores de busca (Busca)

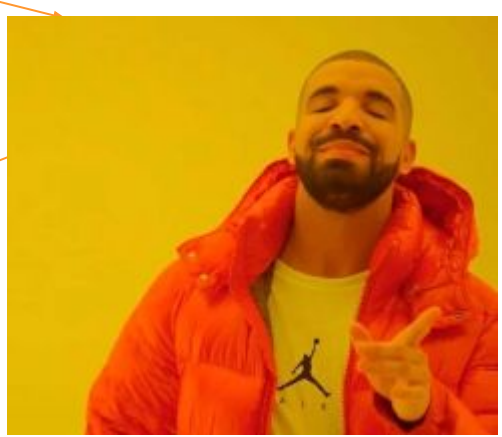
K = 21



Árvores de busca (Busca)

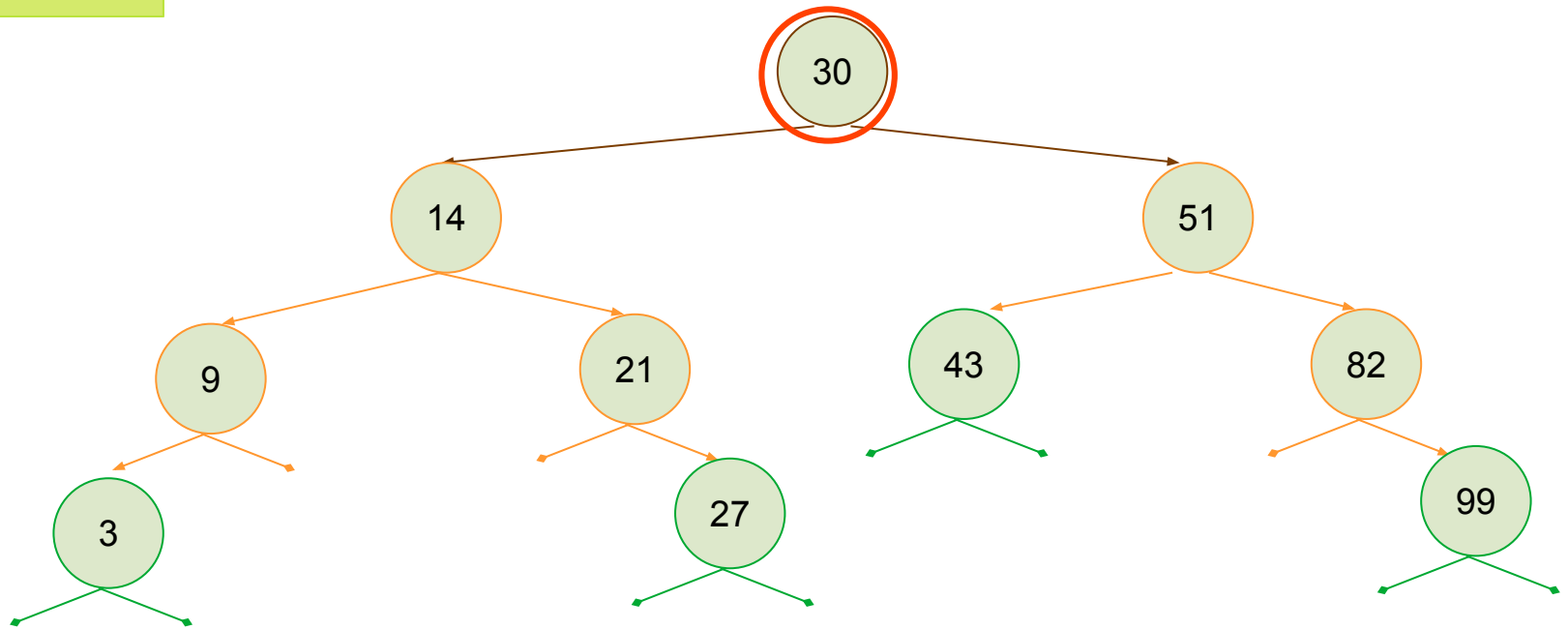
K = 21





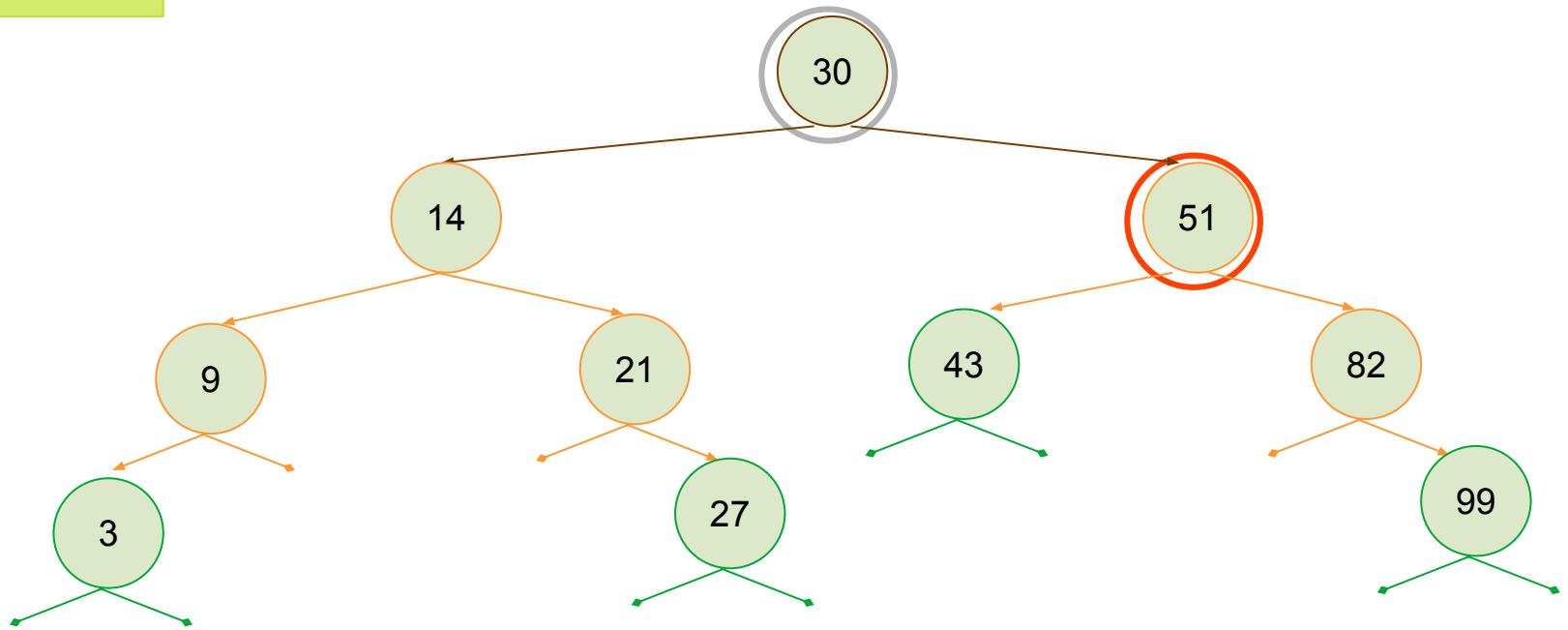
Árvores de busca
(Busca 2)

K = 96



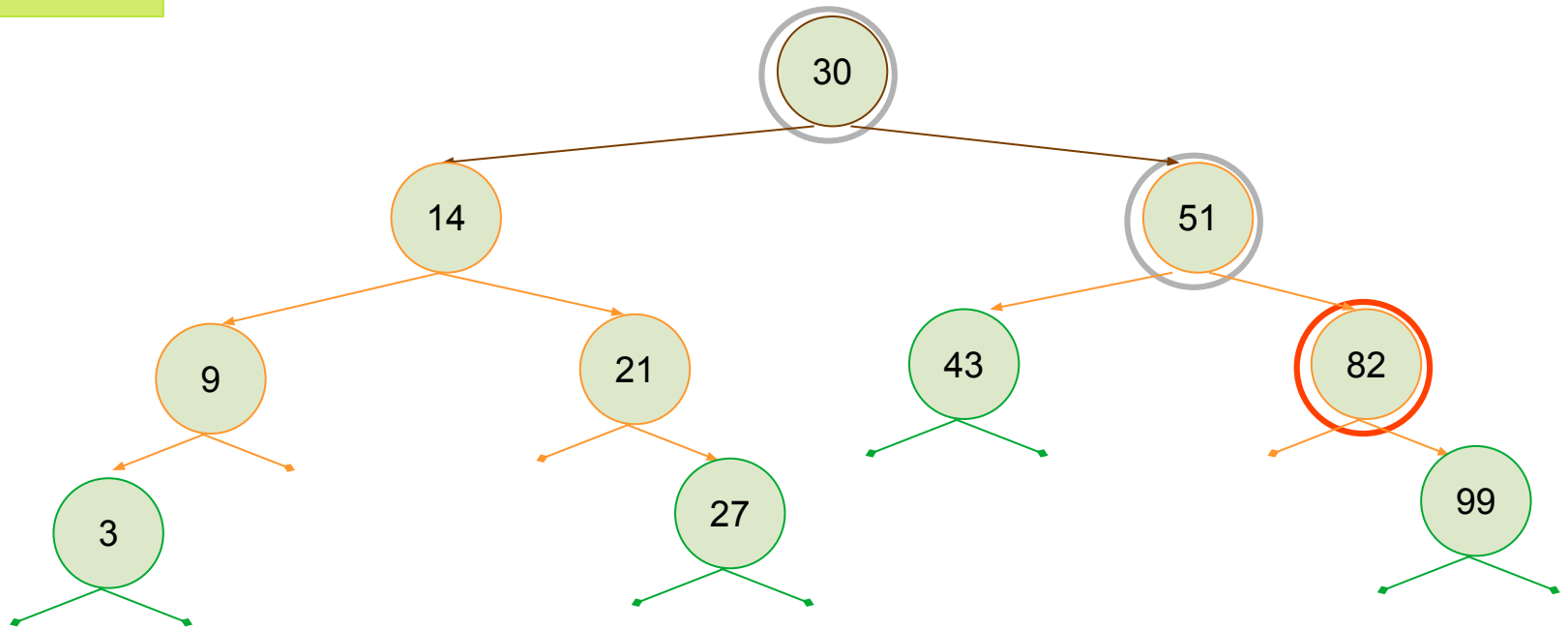
Árvores de busca
(Busca 2)

K = 96



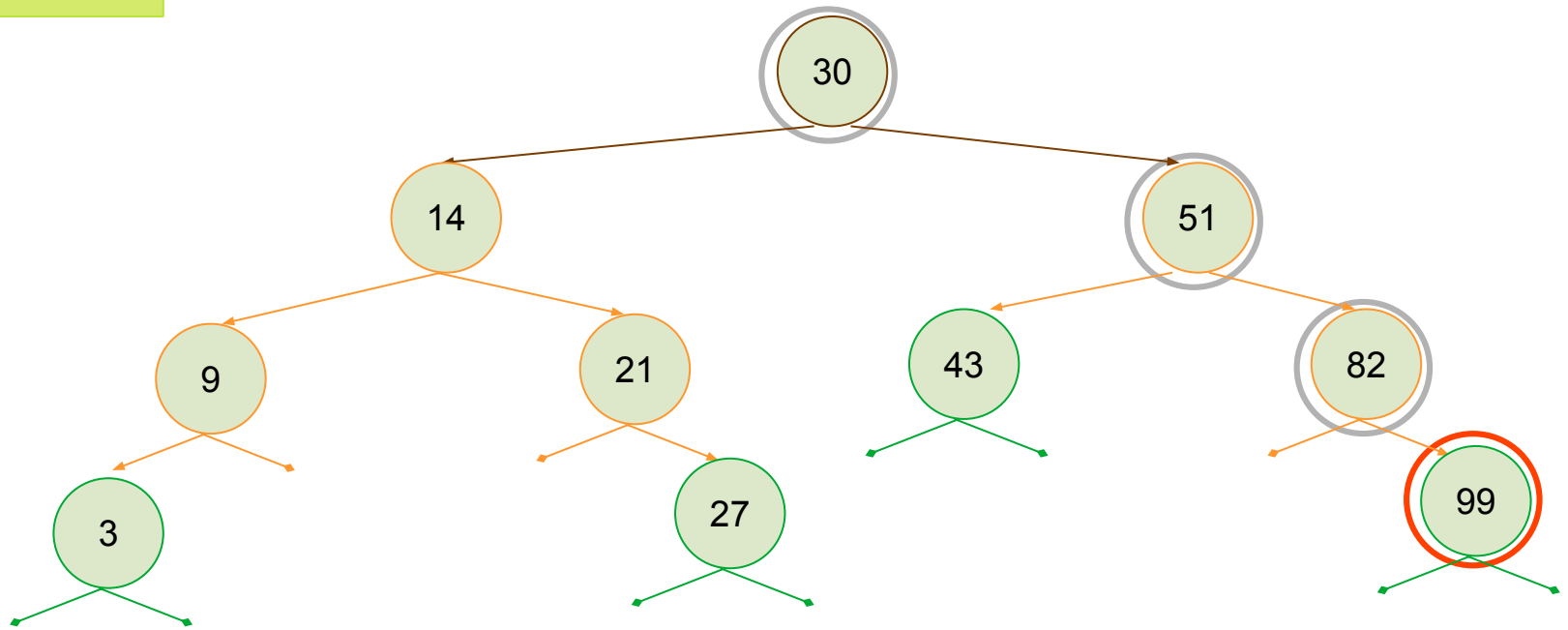
Árvores de busca
(Busca 2)

K = 96



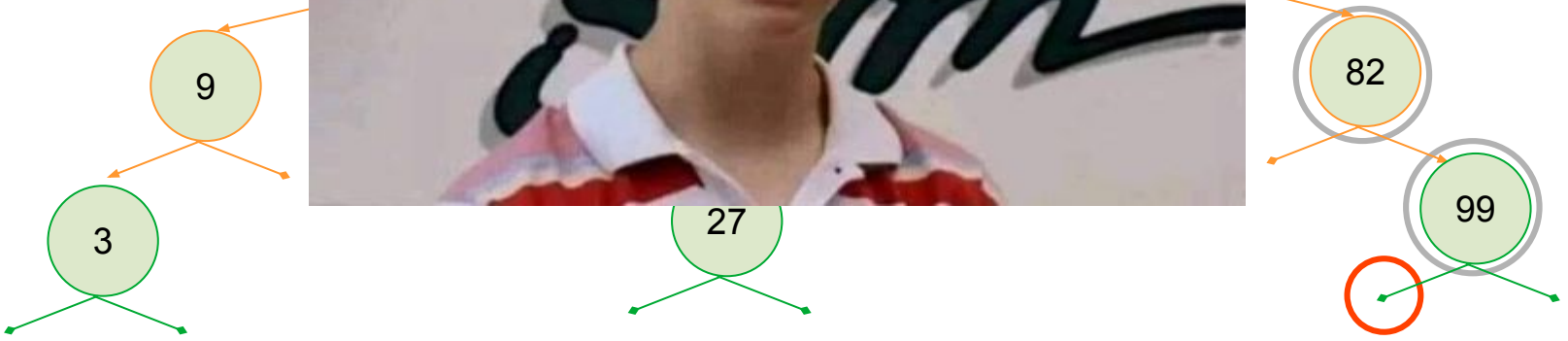
Árvores de busca
(Busca 2)

K = 96



Árvores de busca
(Busca 2)

K = 96



Árvores de busca

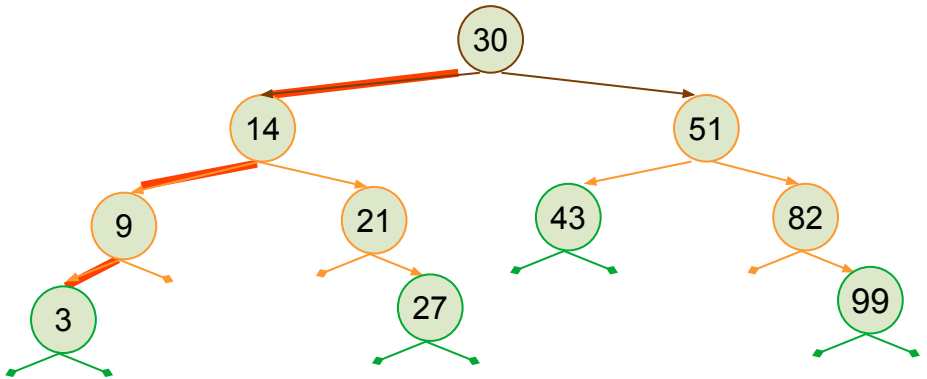
```
Algoritmo buscaChave(Node *raiz, key)
Inicio
```


```
fimAlgoritmo
```

Árvores de busca

- A **altura de um nó** X em uma árvore binária é a distância entre X e o seu descendente mais afastado.
  - Mais precisamente, a altura de X é o número de passos no mais longo caminho que leva de X até uma folha.
  - Os caminhos a que essa definição se refere são os obtido pela iteração das instruções X = X->esq e X = X->dir, em qualquer ordem.
  - A **altura da árvore** é dada pela distância da raiz até as folhas.

Altura = 4

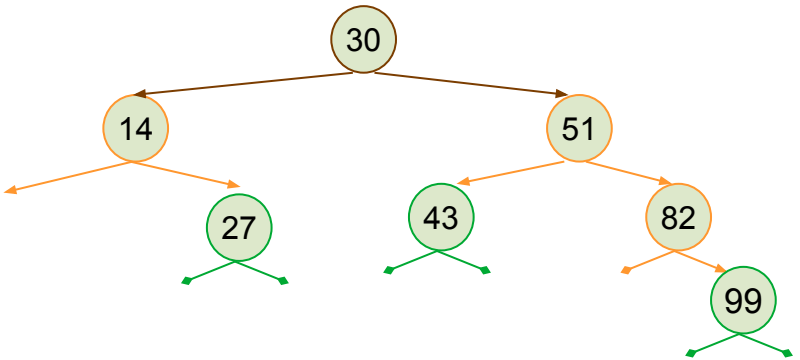


Árvores de busca

- A **altura de um nó** X em uma árvore binária é a distância entre X e o seu descendente mais afastado.
  - Mais precisamente, a altura de X é o número de passos no mais longo caminho que leva de X até uma folha.
  - Os caminhos a que essa definição se refere são os obtido pela iteração das instruções X = X->esq e X = X->dir, em qualquer ordem.
  - A **altura da árvore** é dada pela distância da raiz até as folhas.

Altura = ?

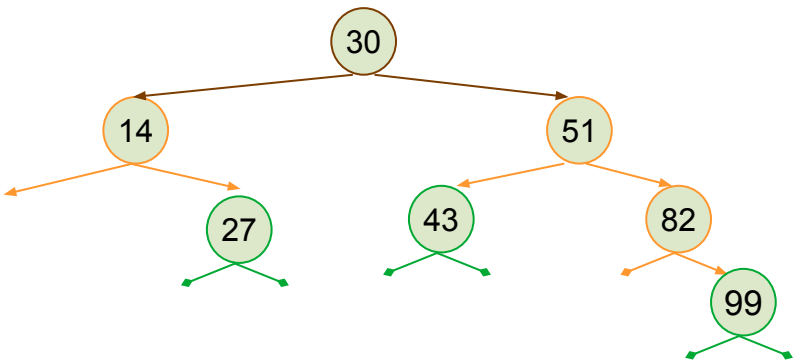
4



Árvores de busca

- A **profundidade** de um nó é a distância deste até a raiz
  - É um forma semelhante ao cálculo da altura, mas agora não consideramos a folha mais afastada e sim o número de passos para chegar do elemento raiz até o elemento procurado.

- Ex. Qual a profundidade do elemento 82?

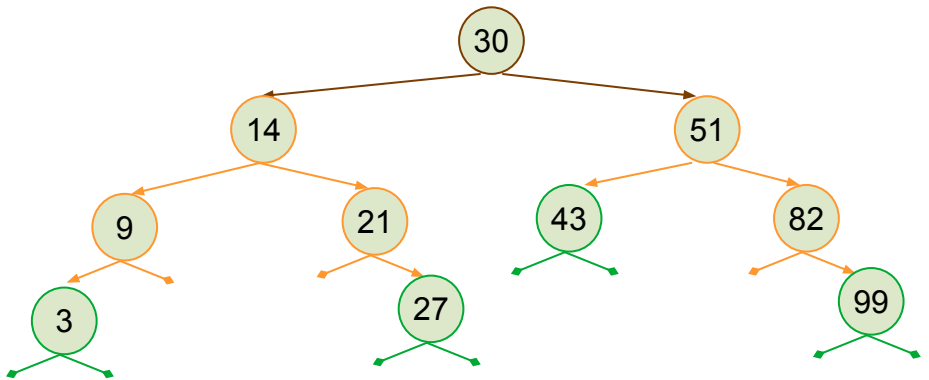


Árvores de busca

- Existem diversas aplicações onde devemos percorrer uma árvore de maneira sistemática, visitando todos os nós da árvore, um a um.
- Existem três formas de percorrer uma árvore binária:
  - Pré-ordem
  - In-ordem
  - Pós-ordem

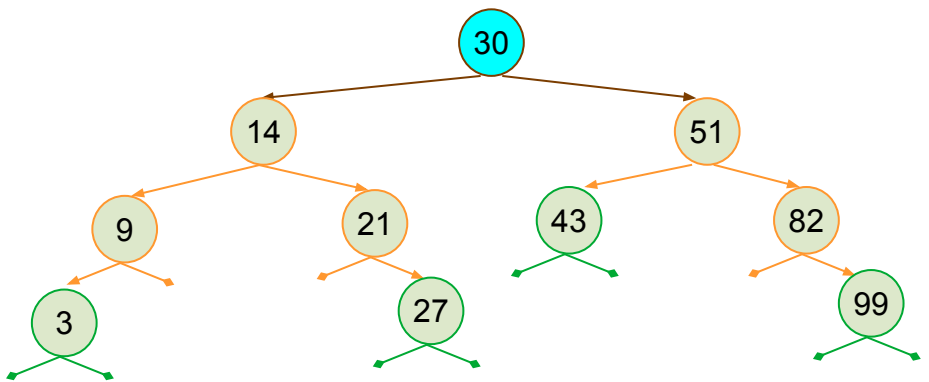
Árvores de busca

- Pré-ordem
  - Visita a raiz
  - Percorre a subárvore esquerda
  - Percorre a subárvore direita



Árvores de busca

- Pré-ordem
  - Visita a raiz
  - Percorre a subárvore esquerda
  - Percorre a subárvore direita

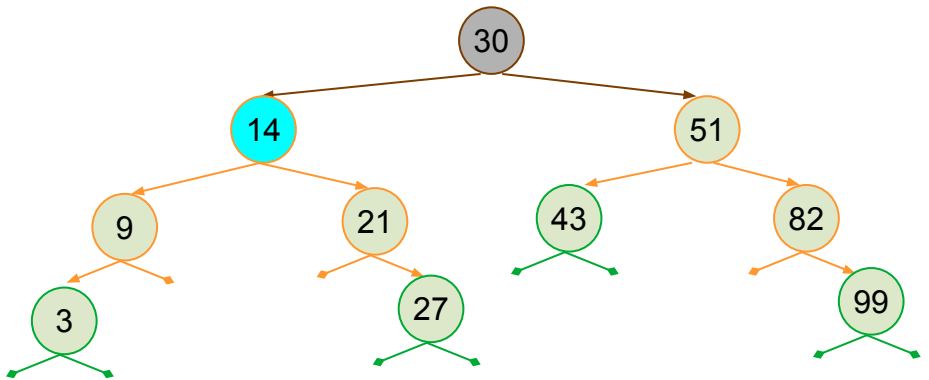


Saída:

30

Árvores de busca

- Pré-ordem
  - Visita a raiz
  - Percorre a subárvore esquerda
  - Percorre a subárvore direita

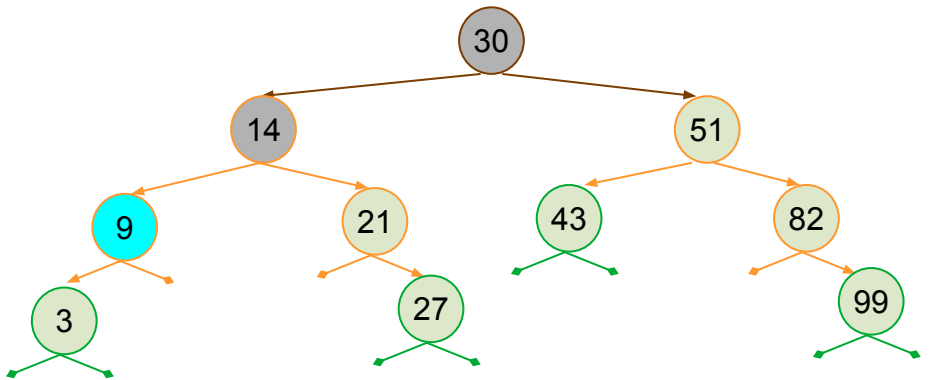


Saída:

30, 14

Árvores de busca

- Pré-ordem
  - Visita a raiz
  - Percorre a subárvore esquerda
  - Percorre a subárvore direita

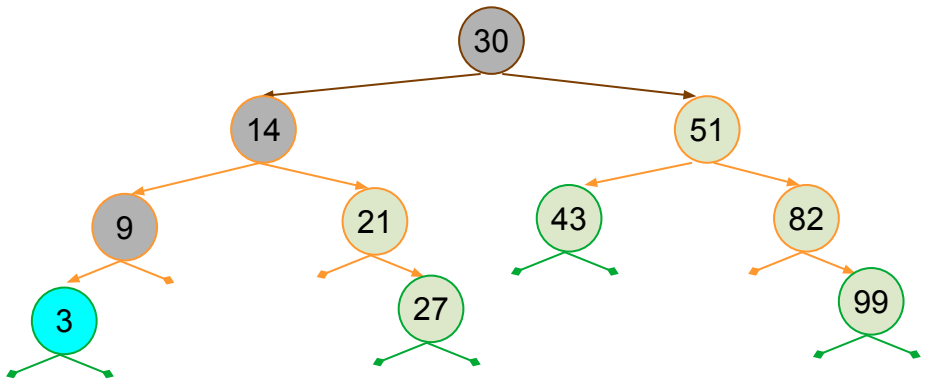


Saída:

30, 14, 9

Árvores de busca

- Pré-ordem
  - Visita a raiz
  - Percorre a subárvore esquerda
  - Percorre a subárvore direita

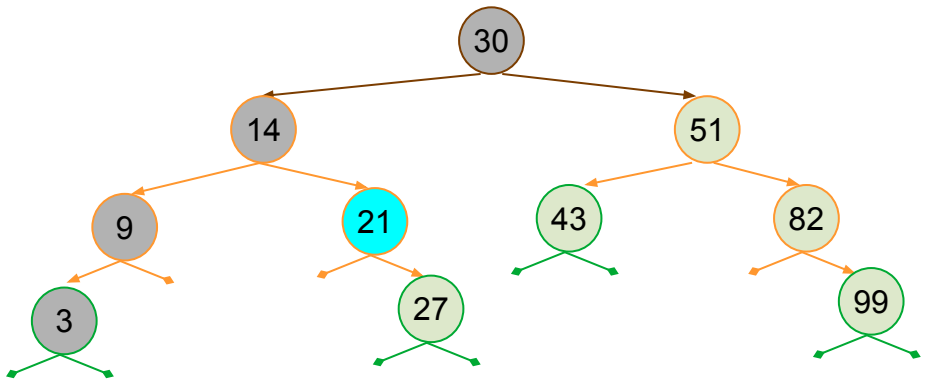


Saída:

30, 14, 9, 3

Árvores de busca

- Pré-ordem
  - Visita a raiz
  - Percorre a subárvore esquerda
  - Percorre a subárvore direita

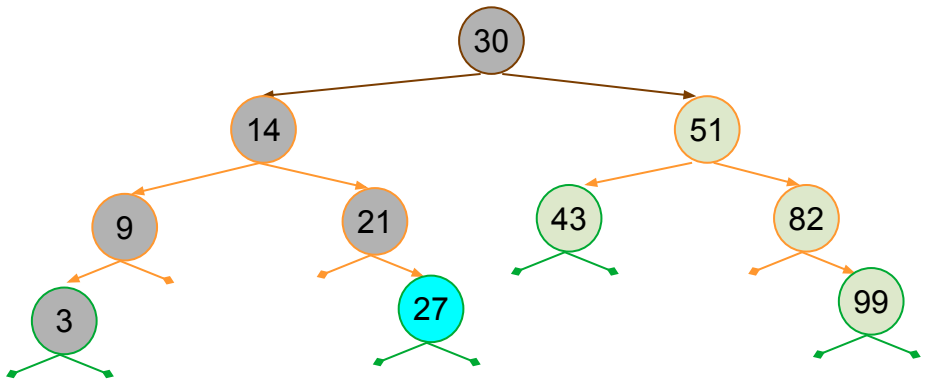


Saída:

30, 14, 9, 3, 21

Árvores de busca

- Pré-ordem
  - Visita a raiz
  - Percorre a subárvore esquerda
  - Percorre a subárvore direita

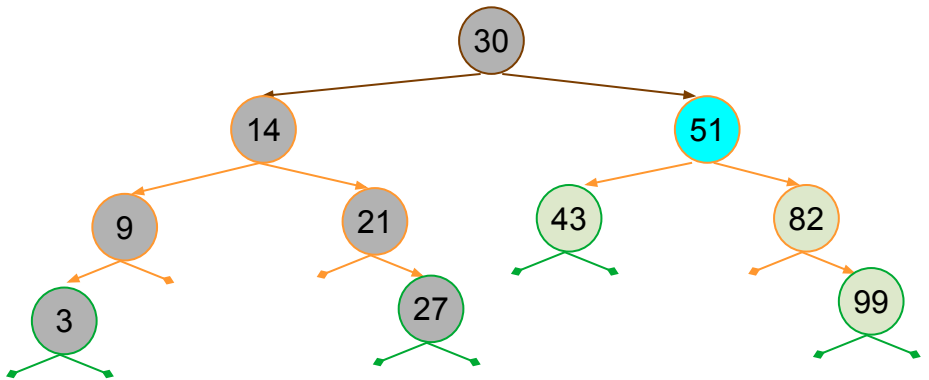


Saída:

30, 14, 9, 3, 21, 27

Árvores de busca

- Pré-ordem
  - Visita a raiz
  - Percorre a subárvore esquerda
  - Percorre a subárvore direita

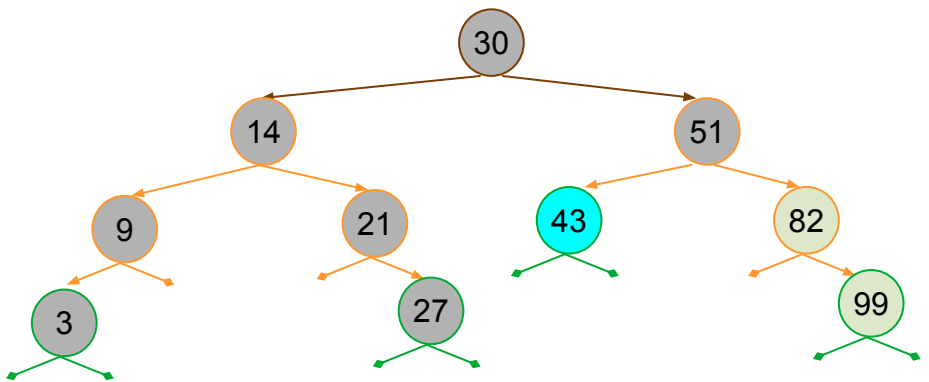


Saída:

30, 14, 9, 3, 21, 27, 51

Árvores de busca

- Pré-ordem
  - Visita a raiz
  - Percorre a subárvore esquerda
  - Percorre a subárvore direita

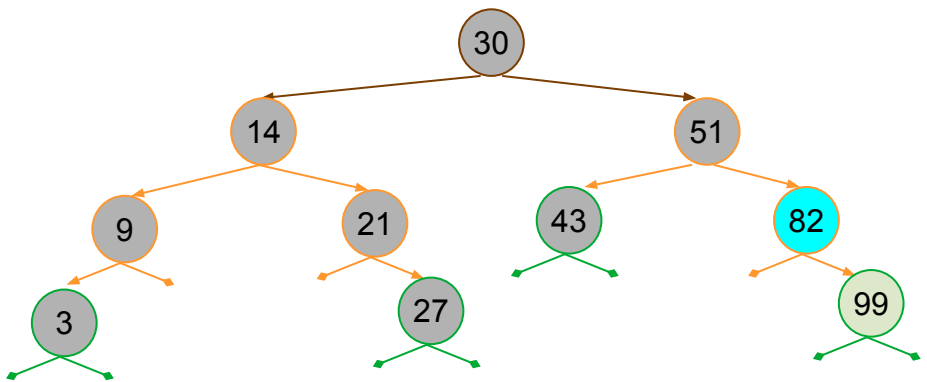


Saída:

30, 14, 9, 3, 21, 27, 51, 43

Árvores de busca

- Pré-ordem
  - Visita a raiz
  - Percorre a subárvore esquerda
  - Percorre a subárvore direita

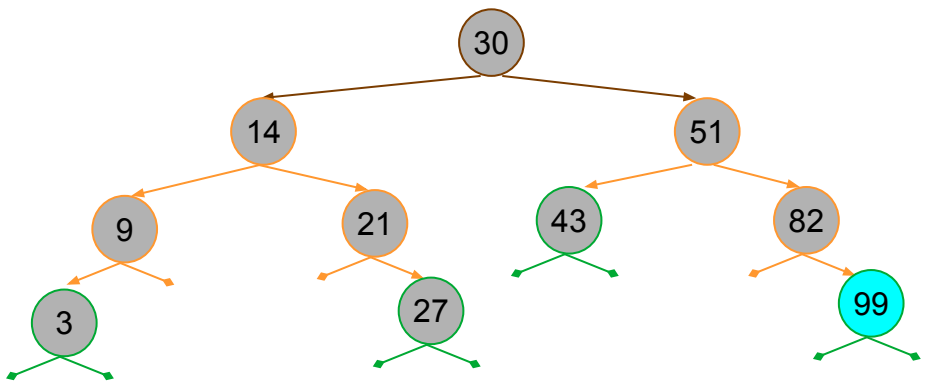


Saída:

30, 14, 9, 3, 21, 27, 51, 43, 82

Árvores de busca

- Pré-ordem
  - Visita a raiz
  - Percorre a subárvore esquerda
  - Percorre a subárvore direita

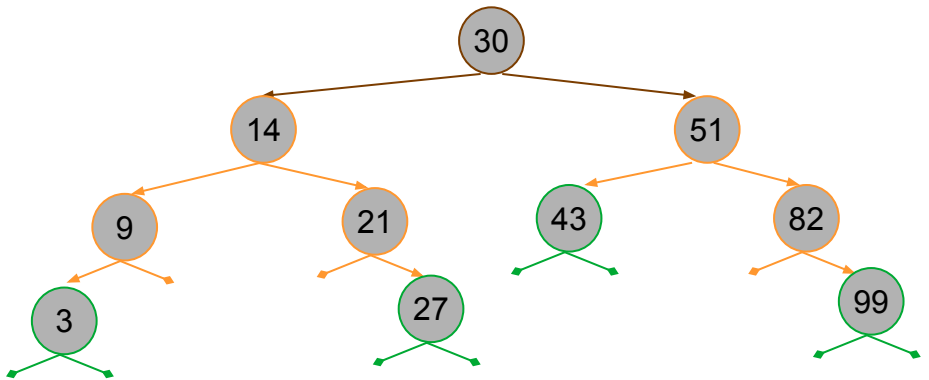


Saída:

30, 14, 9, 3, 21, 27, 51, 43, 82, 99

Árvores de busca

- Pré-ordem
  - Visita a raiz
  - Percorre a subárvore esquerda
  - Percorre a subárvore direita



Saída:

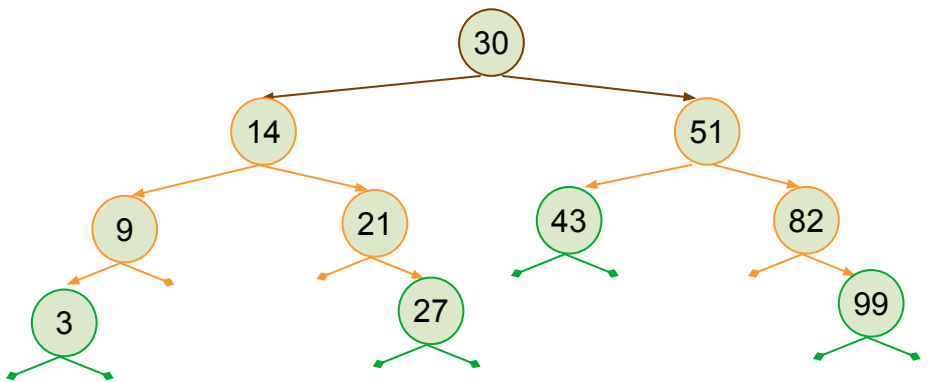
30, 14, 9, 3, 21, 27, 51, 43, 82, 99

Árvores de busca

- Existem diversas aplicações onde devemos percorrer uma árvore de maneira sistemática, visitando todos os nós da árvore, um a um.
- Existem três formas de percorrer uma árvore binária:
  - Pré-ordem
  - **In-ordem**
  - Pós-ordem

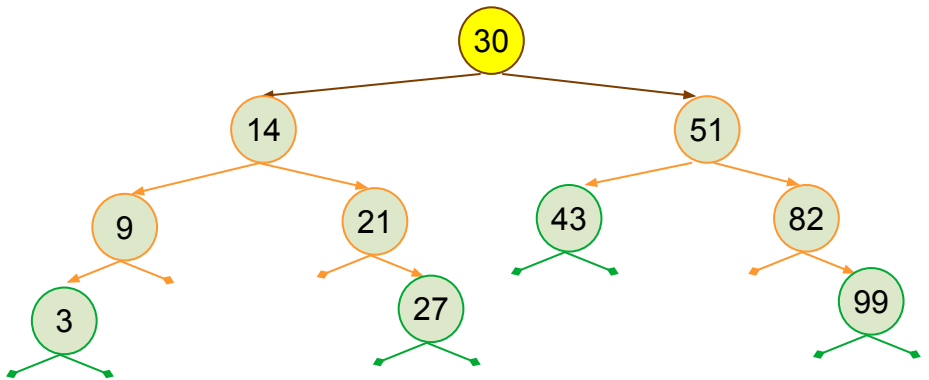
Árvores de busca

- In-ordem
  - Percorre a subárvore esquerda
  - Visita a raiz
  - Percorre a subárvore direita



Árvores de busca

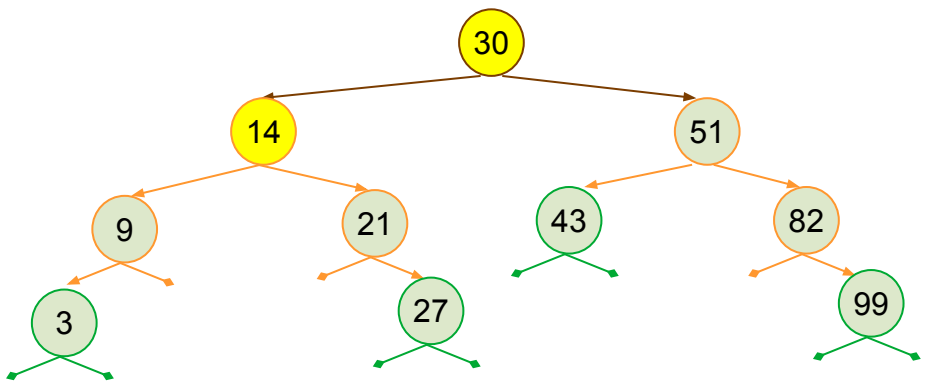
- In-ordem
  - Percorre a subárvore esquerda
  - Visita a raiz
  - Percorre a subárvore direita



Saída:

Árvores de busca

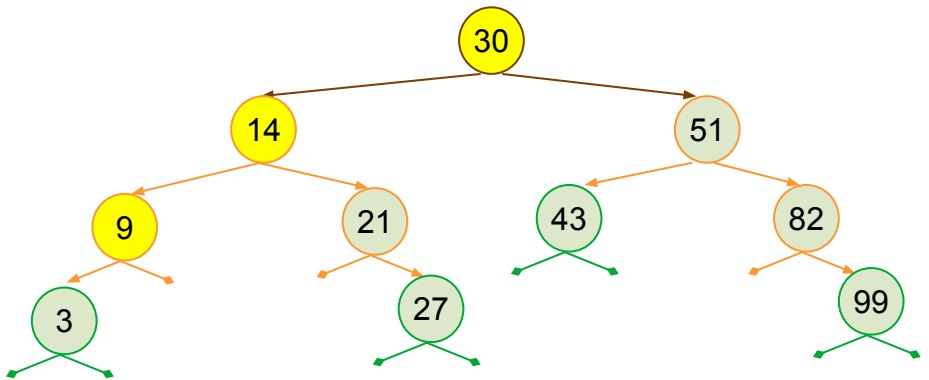
- In-ordem
  - Percorre a subárvore esquerda
  - Visita a raiz
  - Percorre a subárvore direita



Saída:

Árvores de busca

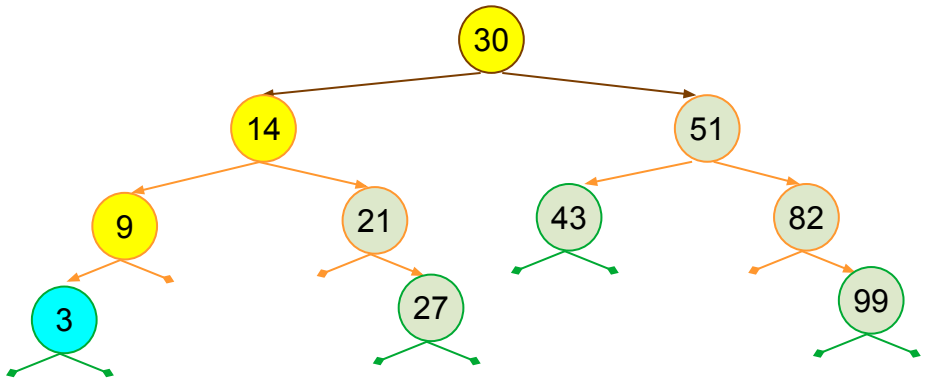
- In-ordem
  - Percorre a subárvore esquerda
  - Visita a raiz
  - Percorre a subárvore direita



Saída:

Árvores de busca

- In-ordem
  - Percorre a subárvore esquerda
  - Visita a raiz
  - Percorre a subárvore direita

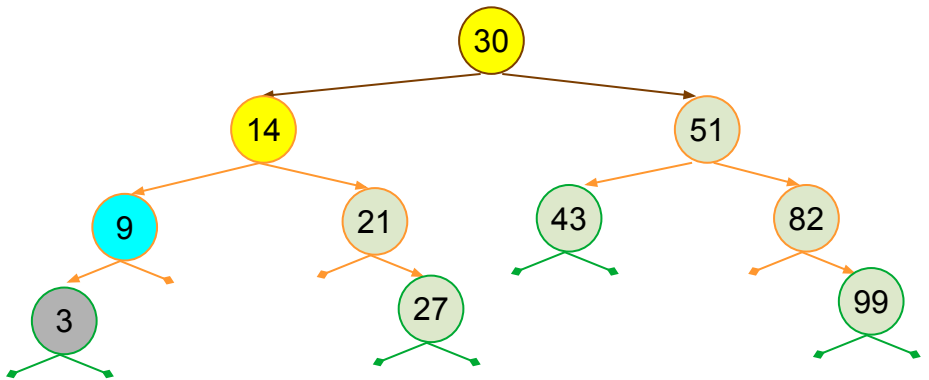


Saída:

3

Árvores de busca

- In-ordem
  - Percorre a subárvore esquerda
  - Visita a raiz
  - Percorre a subárvore direita

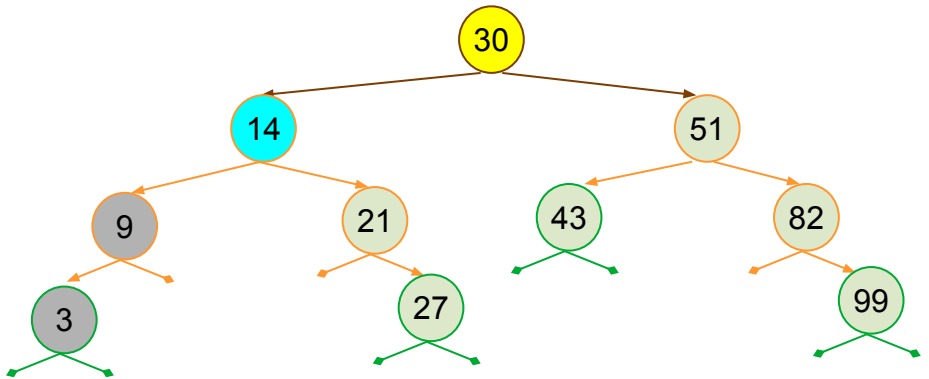


Saída:

3, 9

Árvores de busca

- In-ordem
  - Percorre a subárvore esquerda
  - Visita a raiz
  - Percorre a subárvore direita

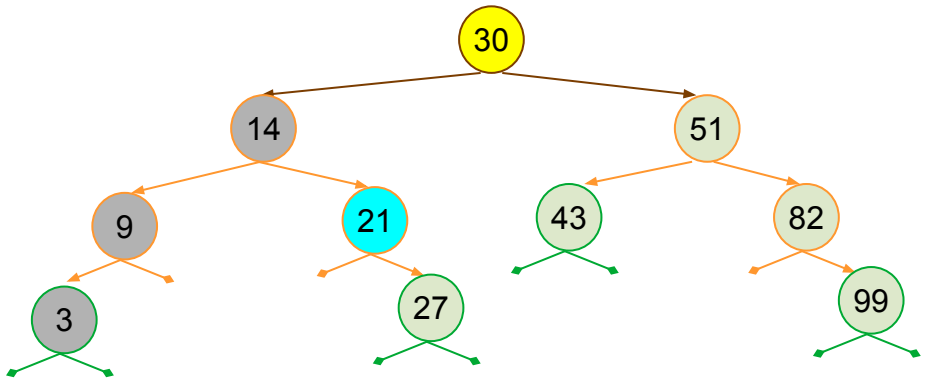


Saída:

3, 9, 14

Árvores de busca

- In-ordem
  - Percorre a subárvore esquerda
  - Visita a raiz
  - Percorre a subárvore direita

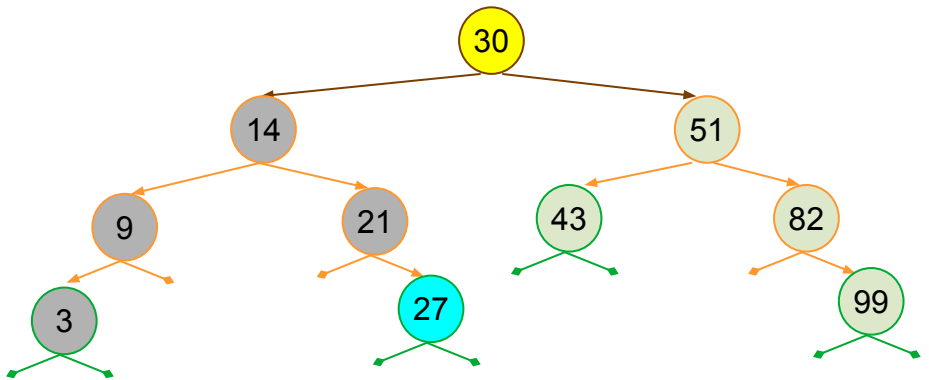


Saída:

3, 9, 14, 21

Árvores de busca

- In-ordem
  - Percorre a subárvore esquerda
  - Visita a raiz
  - Percorre a subárvore direita

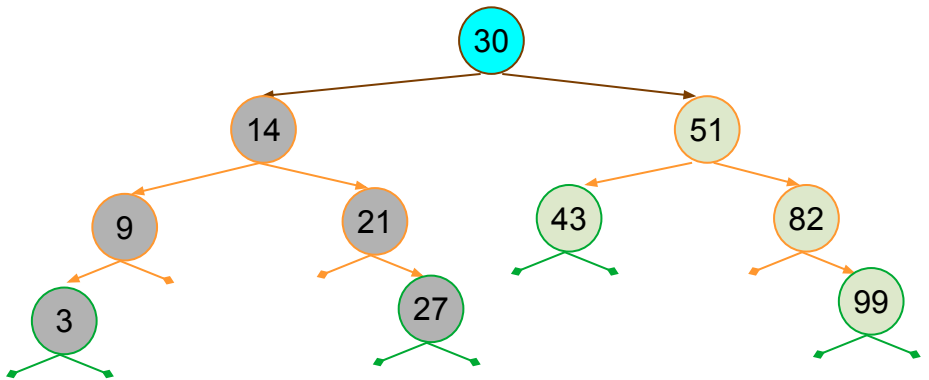


Saída:

3, 9, 14, 21, 27

Árvores de busca

- In-ordem
  - Percorre a subárvore esquerda
  - Visita a raiz
  - Percorre a subárvore direita

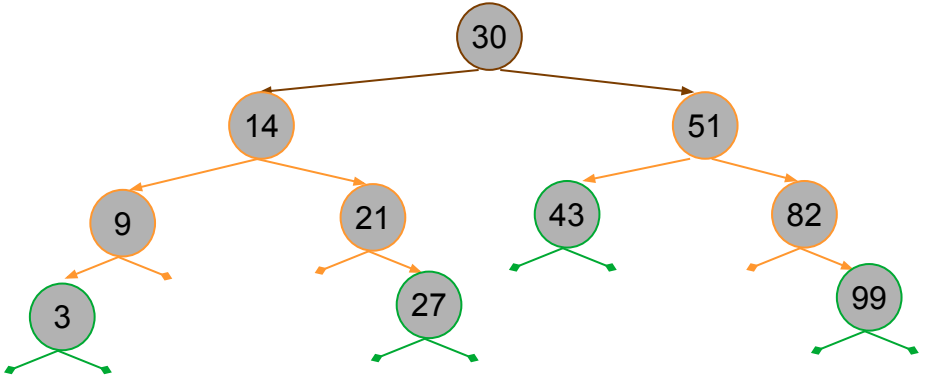


Saída:

3, 9, 14, 21, 27, 30

Árvores de busca

- In-ordem
  - Percorre a subárvore esquerda
  - Visita a raiz
  - Percorre a subárvore direita



Saída:

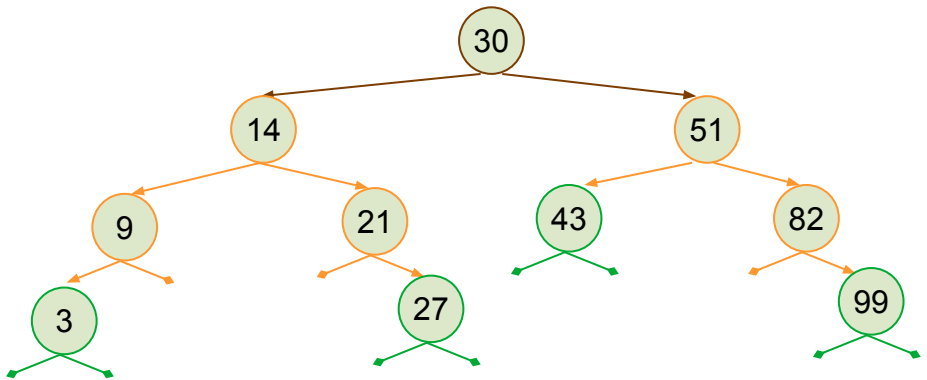
3, 9, 14, 21, 27, 30, 43, 51, 82, 99

Árvores de busca

- Existem diversas aplicações onde devemos percorrer uma árvore de maneira sistemática, visitando todos os nós da árvore, um a um.
- Existem três formas de percorrer uma árvore binária:
  - Pré-ordem
  - In-ordem
  - **Pós-ordem**

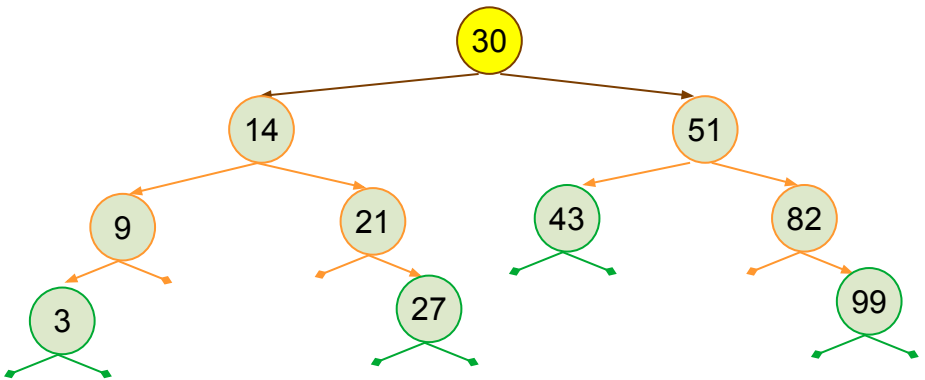
Árvores de busca

- Pós-ordem
  - Percorre a subárvore esquerda
  - Percorre a subárvore direita
  - Visita a raiz



Árvores de busca

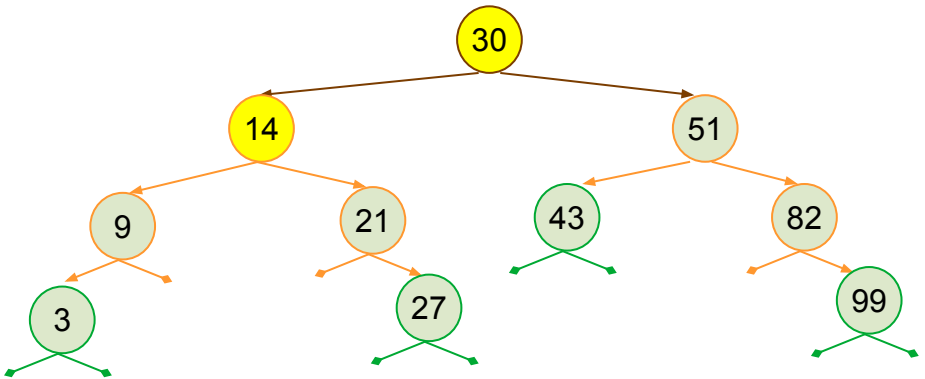
- Pós-ordem
  - Percorre a subárvore esquerda
  - Percorre a subárvore direita
  - Visita a raiz



Saída:

Árvores de busca

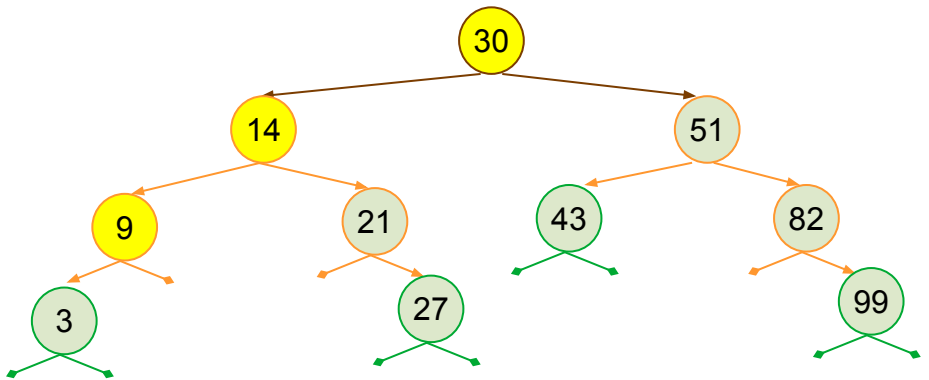
- Pós-ordem
  - Percorre a subárvore esquerda
  - Percorre a subárvore direita
  - Visita a raiz



Saída:

Árvores de busca

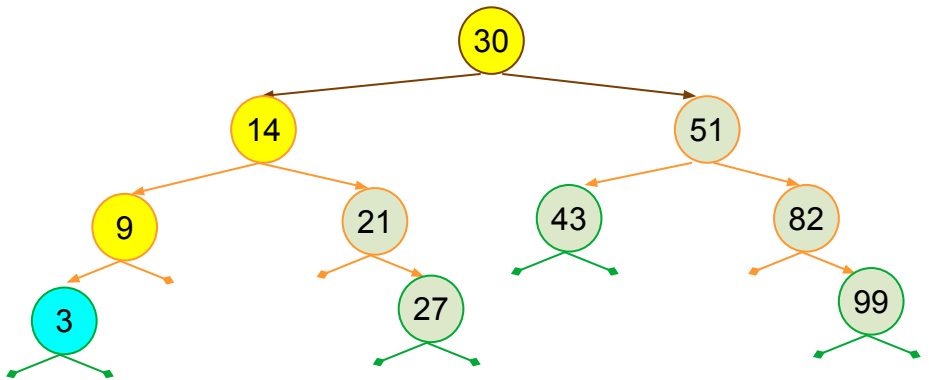
- Pós-ordem
  - Percorre a subárvore esquerda
  - Percorre a subárvore direita
  - Visita a raiz



Saída:

Árvores de busca

- Pós-ordem
  - Percorre a subárvore esquerda
  - Percorre a subárvore direita
  - Visita a raiz

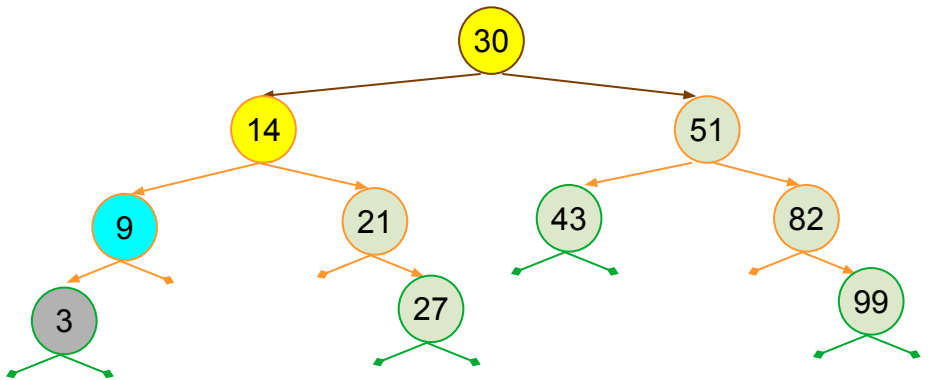


Saída:

3

Árvores de busca

- Pós-ordem
  - Percorre a subárvore esquerda
  - Percorre a subárvore direita
  - Visita a raiz

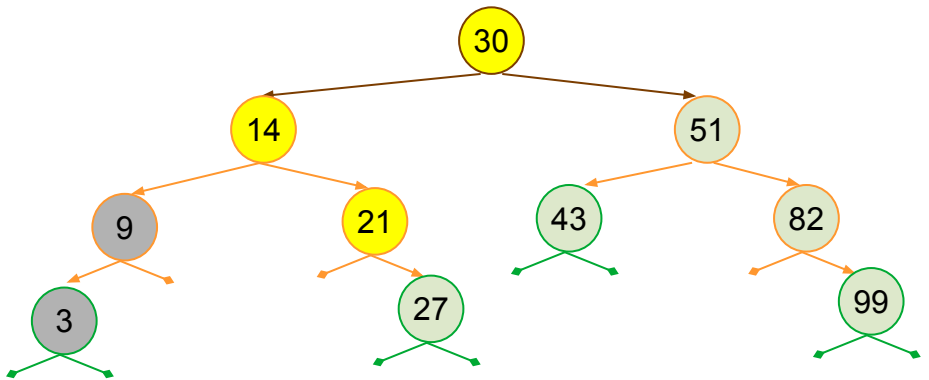


Saída:

3, 9

Árvores de busca

- Pós-ordem
  - Percorre a subárvore esquerda
  - Percorre a subárvore direita
  - Visita a raiz

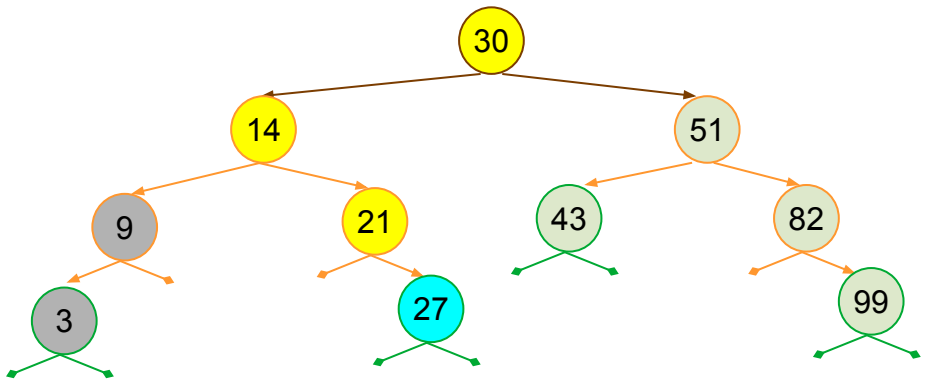


Saída:

3, 9

Árvores de busca

- Pós-ordem
  - Percorre a subárvore esquerda
  - Percorre a subárvore direita
  - Visita a raiz

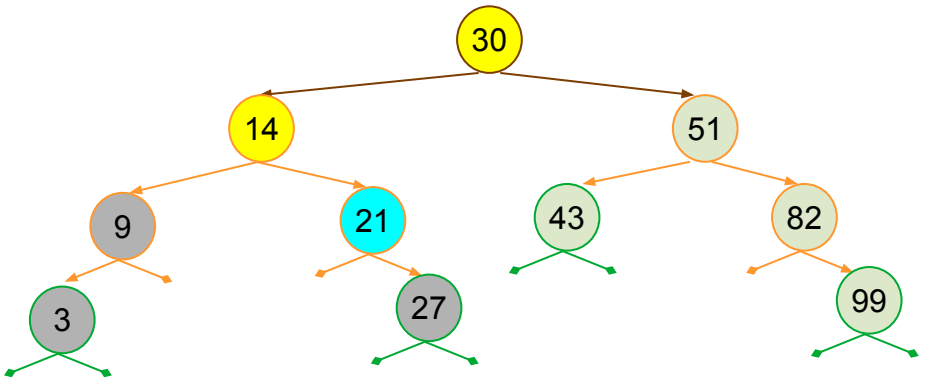


Saída:

3, 9, 27

Árvores de busca

- Pós-ordem
  - Percorre a subárvore esquerda
  - Percorre a subárvore direita
  - Visita a raiz

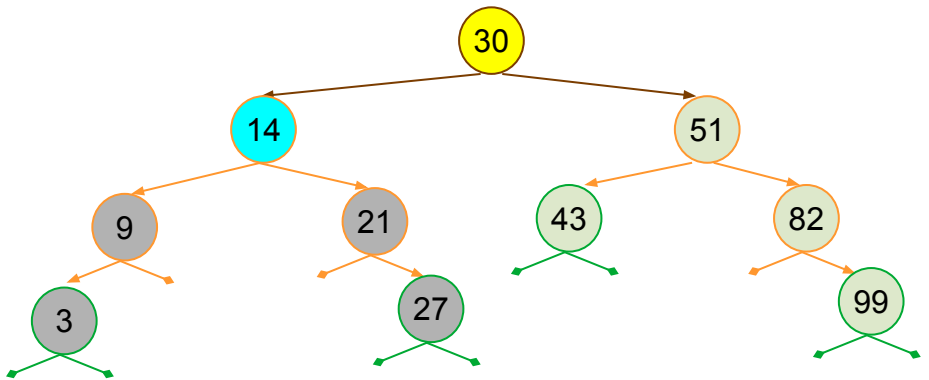


Saída:

3, 9, 27, 21,

Árvores de busca

- Pós-ordem
  - Percorre a subárvore esquerda
  - Percorre a subárvore direita
  - Visita a raiz

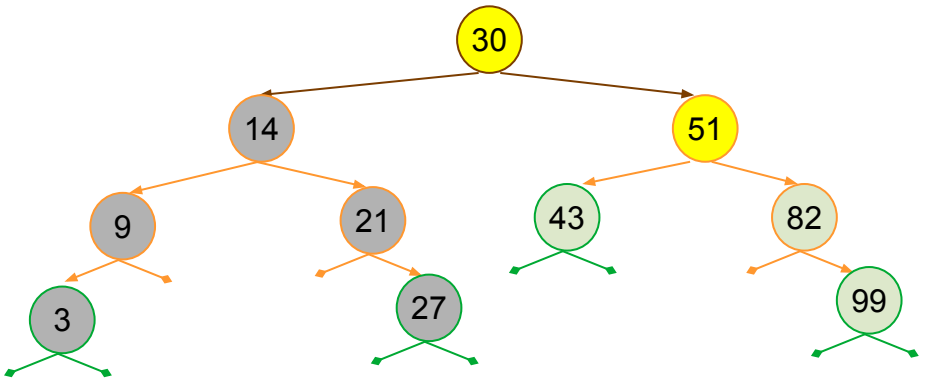


Saída:

3, 9, 27, 21, 14

Árvores de busca

- Pós-ordem
  - Percorre a subárvore esquerda
  - Percorre a subárvore direita
  - Visita a raiz

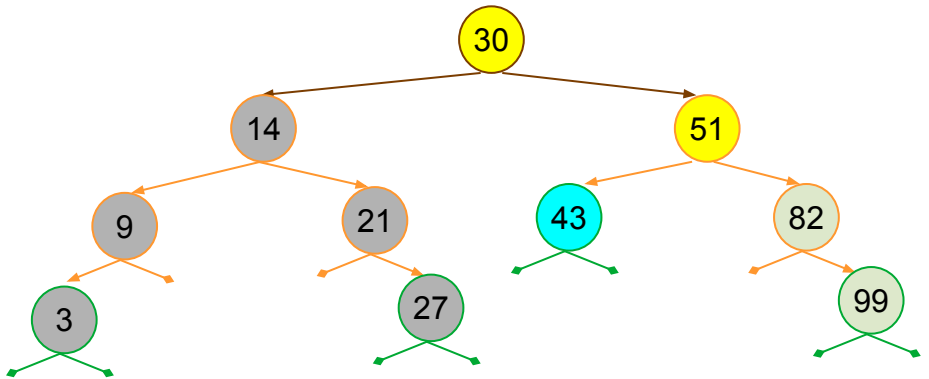


Saída:

3, 9, 27, 21, 14

Árvores de busca

- Pós-ordem
  - Percorre a subárvore esquerda
  - Percorre a subárvore direita
  - Visita a raiz

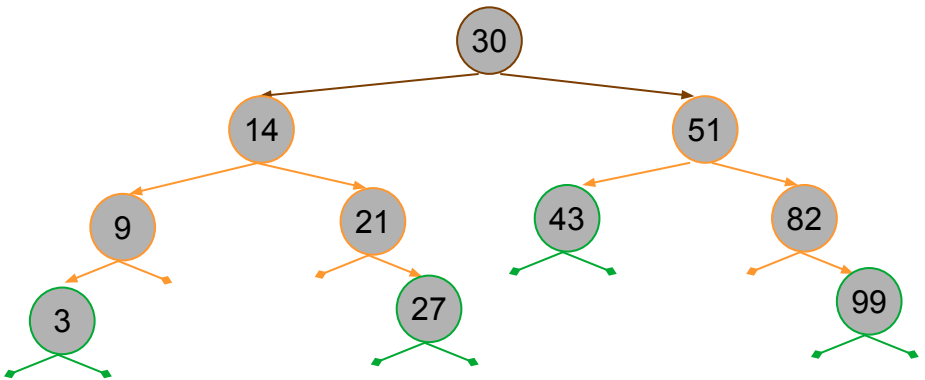


Saída:

3, 9, 27, 21, 14, 43

Árvores de busca

- Pós-ordem
  - Percorre a subárvore esquerda
  - Percorre a subárvore direita
  - Visita a raiz



Saída:

3, 9, 27, 21, 14, 43 , 99, 82, 51, 30

Árvores de busca

- Árvores representam um ótima otimização de performance para o acesso a informações
  - A complexidade é da ordem logarítmica

Árvores de busca

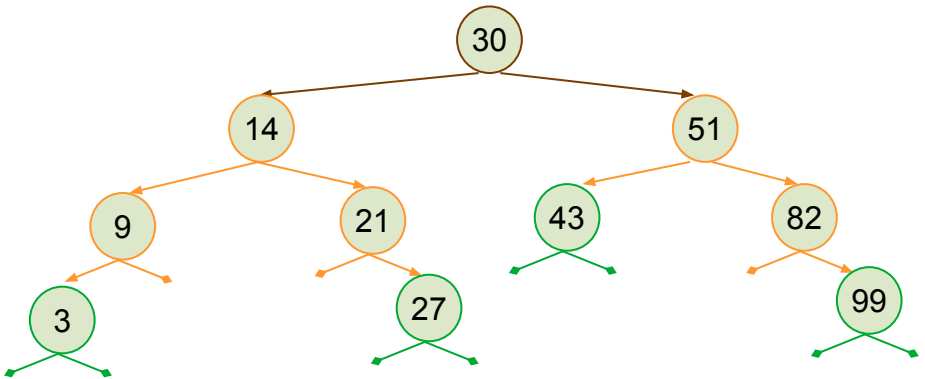
- Árvores representam um ótima otimização de performance para o acesso a informações
  - A complexidade é da ordem logarítmica

Árvores de busca

- Árvores representam um ótima otimização de performance para o acesso a informações
  - A complexidade é difícil e depende da organização da árvore
    - Árvores balanceadas mantém uma complexidade de acesso na ordem de $\log_b n$, pois mantém seu crescimento controlado
  - Para árvores desbalanceadas a complexidade pode chegar a ser linear, no caso de uma árvore degenerada.

Árvores de busca

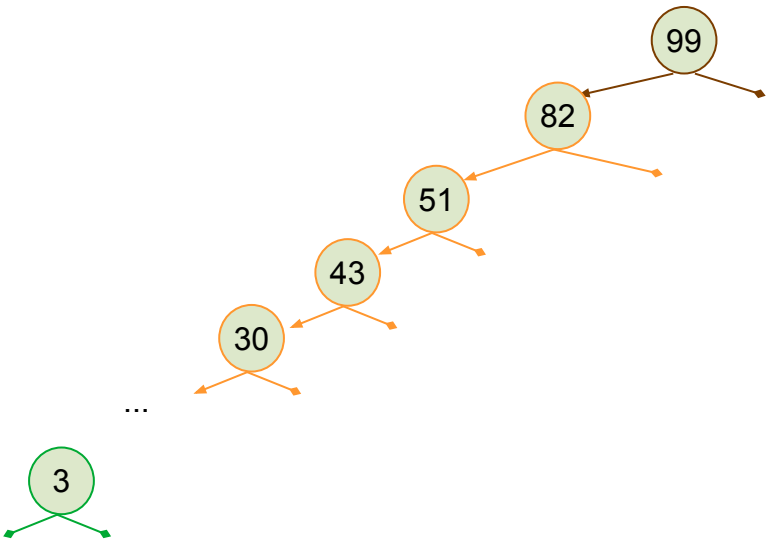
- Árvore balanceada



Ordem de inserção dos elementos:
30, 51, 14, 43, 82, 21, 9, 3, 27, 99

Árvores de busca

- Árvore degenerada



Ordem de inserção dos elementos:
99, 82, 51, 43, 30, 27, 21, 14, 9, 3

https://codeshare.io/mpRgdb