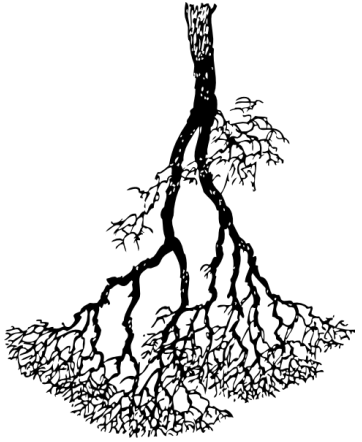
Árvores Binárias de Busca (BST)



Árvores Binária

- As árvores binárias são árvores que os nós possuem no máxima 2 filhos
- Deve possuir pelo menos 3 atributos:

```
1   typedef struct node {
2       int value;
3       struct node *left, *right;
4   } Node;
```

- O campo int value, apenas ilustra um tipo de dados simples
  - Você pode substituí-lo por qualquer tipo que julgar necessário: char, double, struct…

Árvores Binária

- A inserção deve seguir a forma das listas:
  - Aloca memória
  - Atribui o valor para os campos de armazenamento
  - NULLifica os ponteiros que apontam para esquerda e direita
  - Encontra a melhor posição na árvore para o novo nó

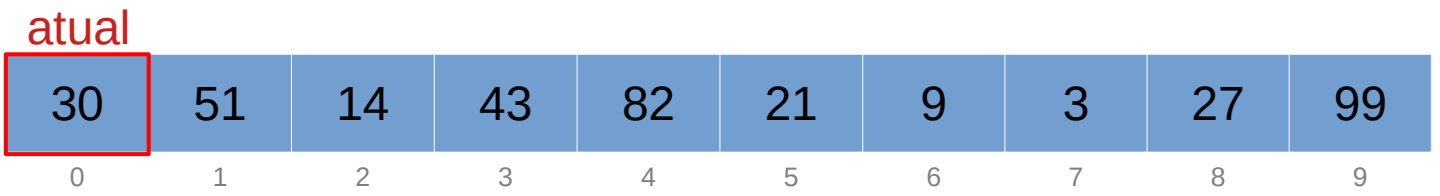
```
newnodo=(Node *) malloc(sizeof(Node));
newnodo->data=data
newnodo->left=NULL;
newnodo->right=NULL;
```

```
1   typedef struct node {
2       int value;
3       struct node *left, *right;
4   } Node;
```

Árvores de busca

- Uma árvore binária de busca é uma árvore binária que restringe a forma que os nós são inseridos:
  - Binary Search Tree (BST)
- Para todo nó de uma árvore de busca binária:
  - O elemento à esquerda deve ser menor ou igual ao pai
  - O elemento à direita deve ser maior ou igual ao pai
  - A igualdade deve ser escolhida ou para a direita ou para a esquerda

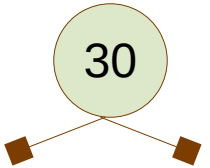
Árvores de busca (Ex)



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

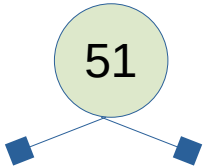


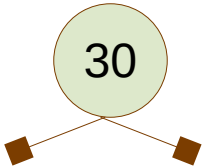
Raiz

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



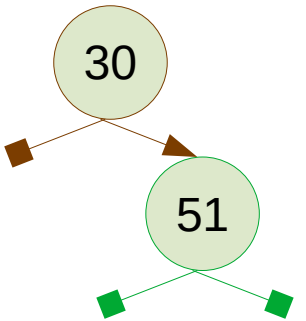


Raiz

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

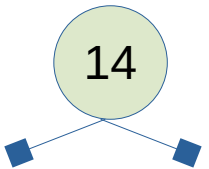


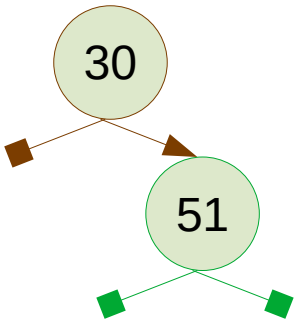
Raiz

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



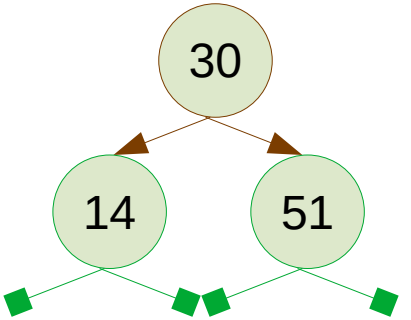


Raiz

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

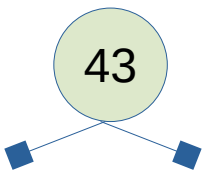


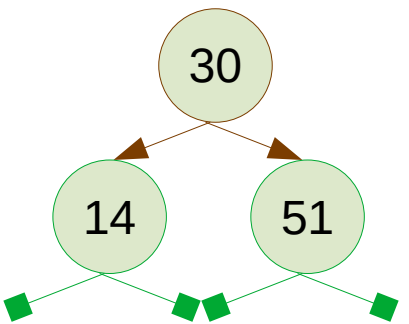
Raiz

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



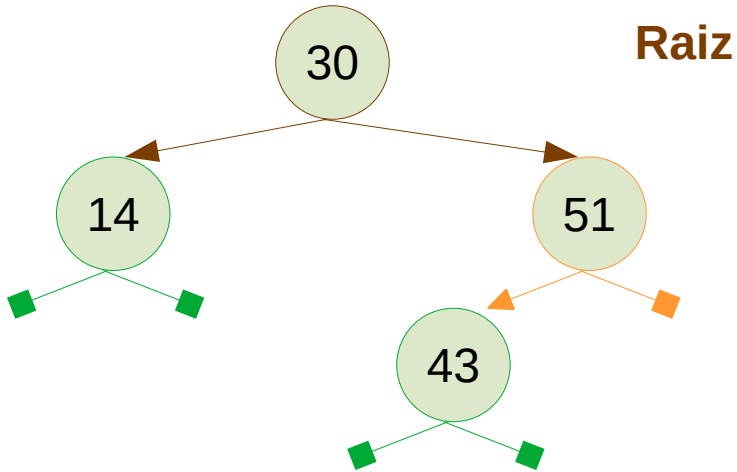


Raiz

Árvores de busca (Ex)

atual

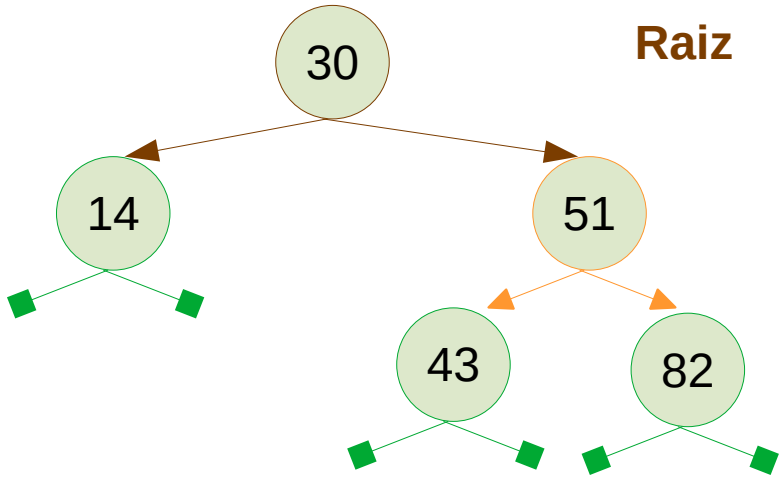
30	51	14	43	82	21	9	3	27	99
0	1	2	3	4	5	6	7	8	9



Árvores de busca (Ex)

atual

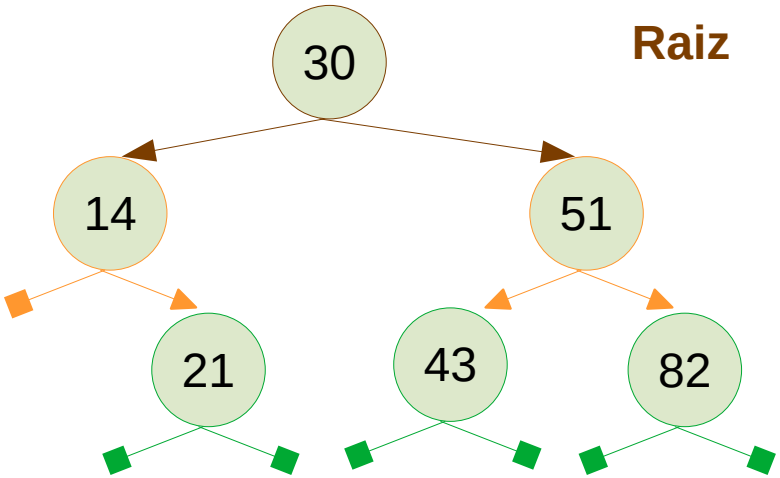
30	51	14	43	82	21	9	3	27	99
0	1	2	3	4	5	6	7	8	9



Árvores de busca (Ex)

atual

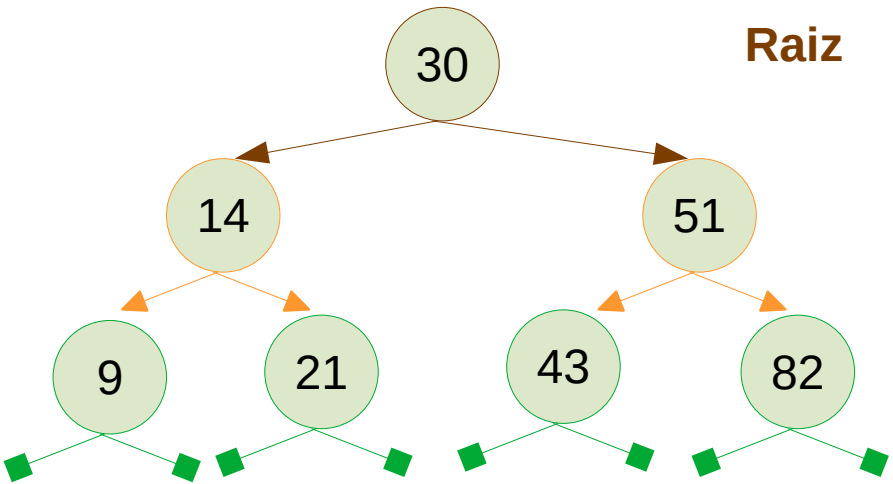
30	51	14	43	82	21	9	3	27	99
0	1	2	3	4	5	6	7	8	9



Árvores de busca (Ex)

atual

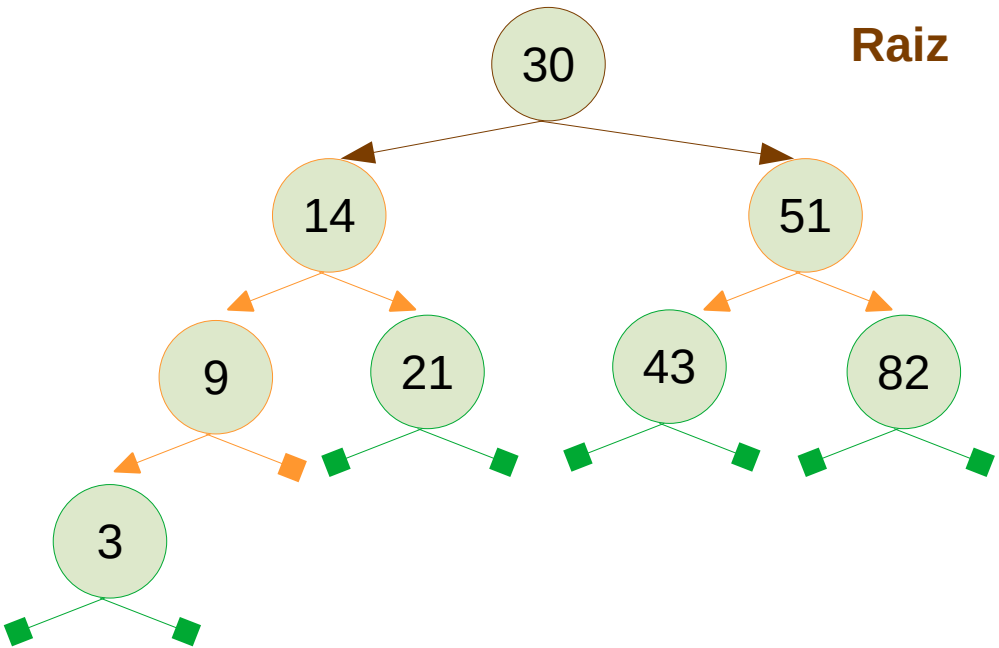
30	51	14	43	82	21	9	3	27	99
0	1	2	3	4	5	6	7	8	9



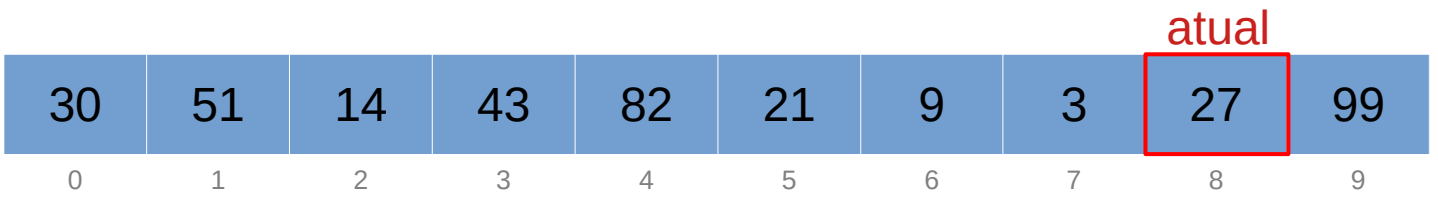
Árvores de busca (Ex)

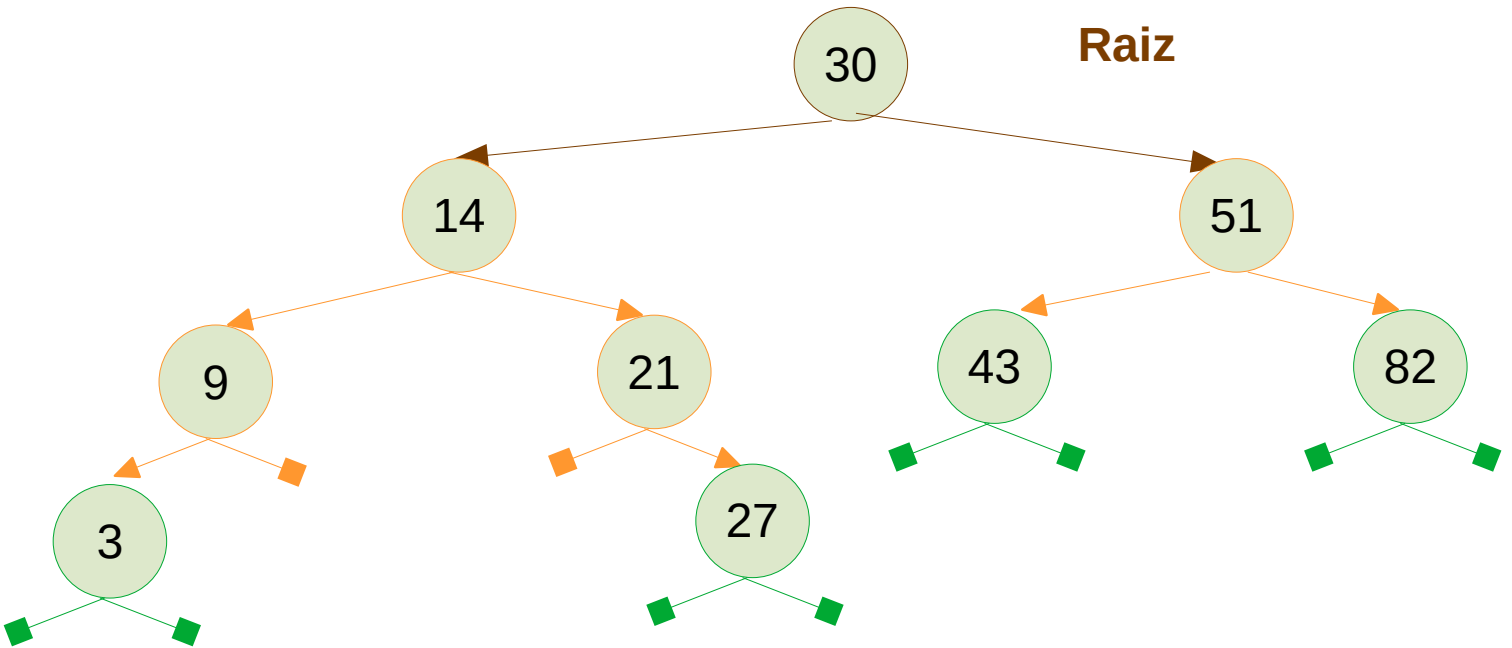
atual

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

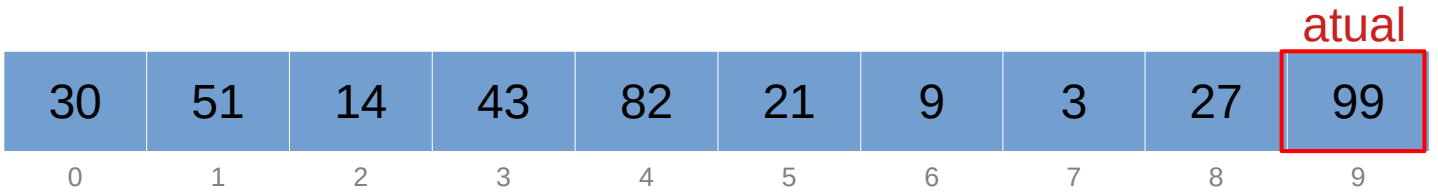


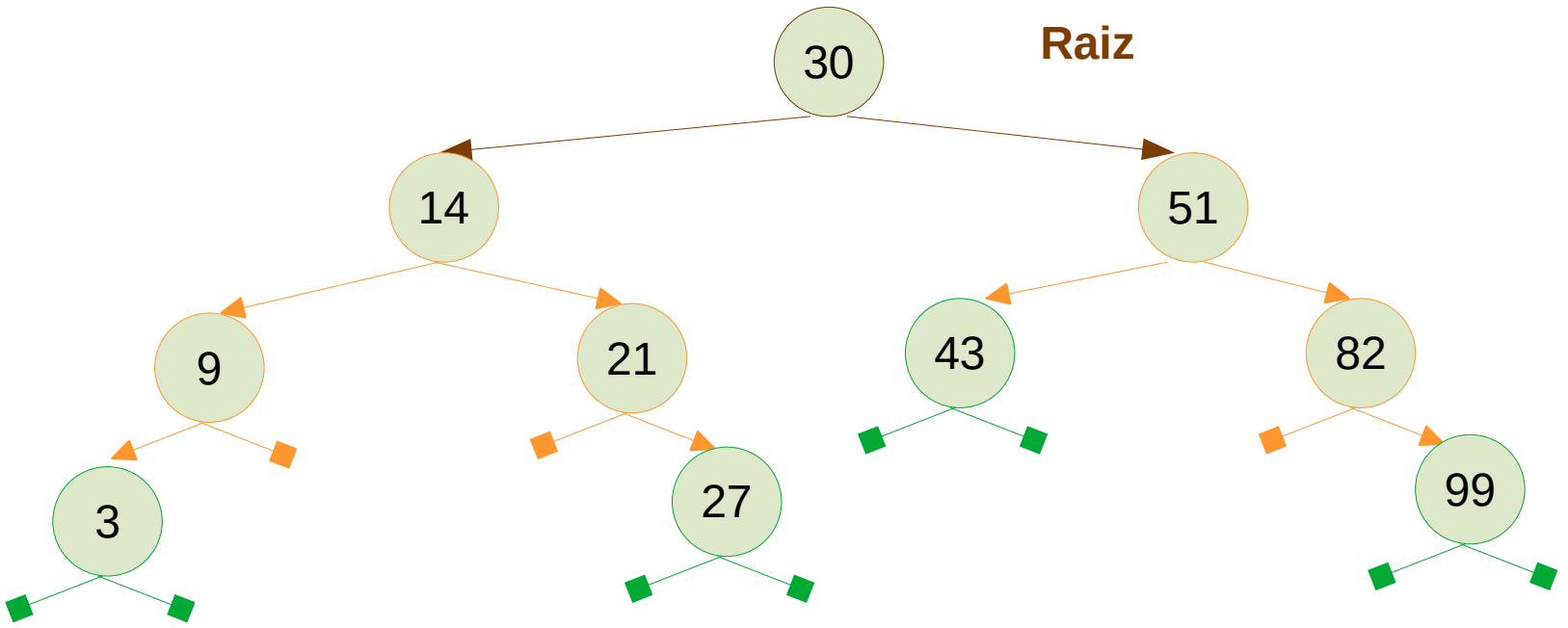
Árvores de busca (Ex)





Árvores de busca (Ex)



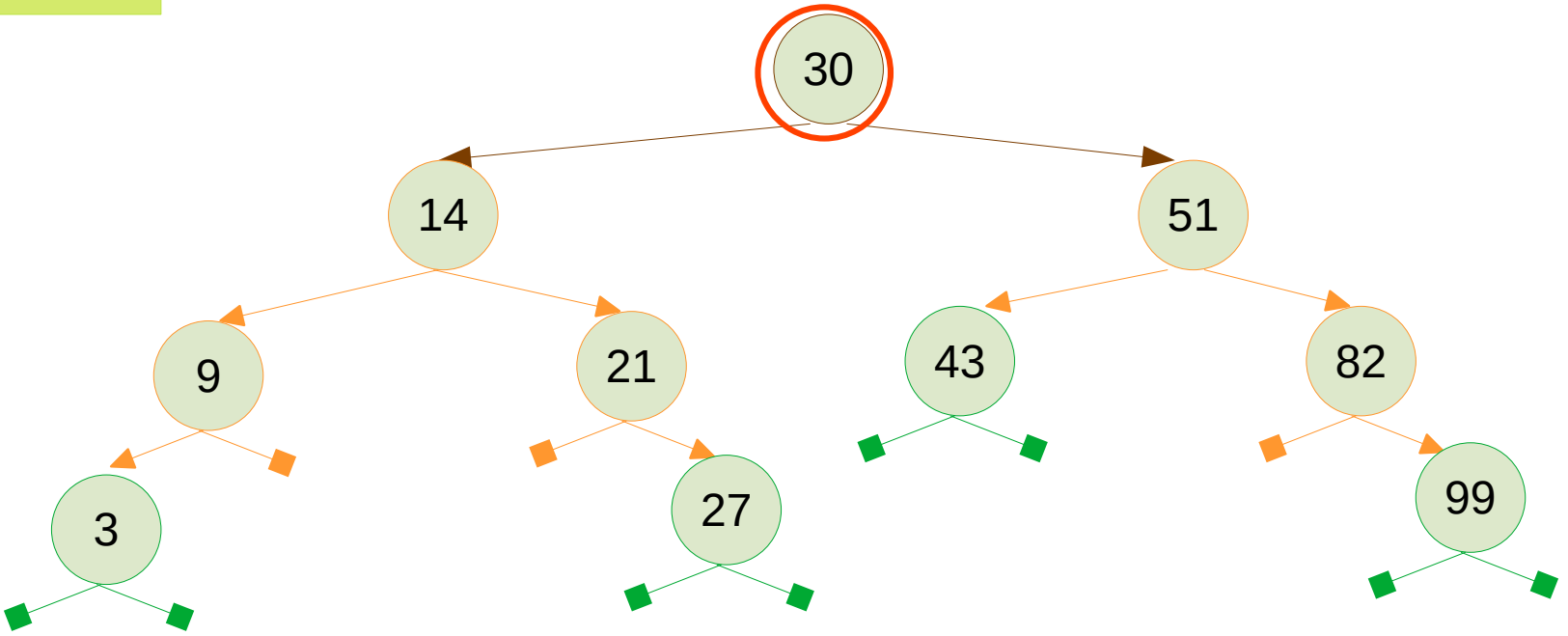


Árvores de busca

```
Algoritmo adcionarNo(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
```

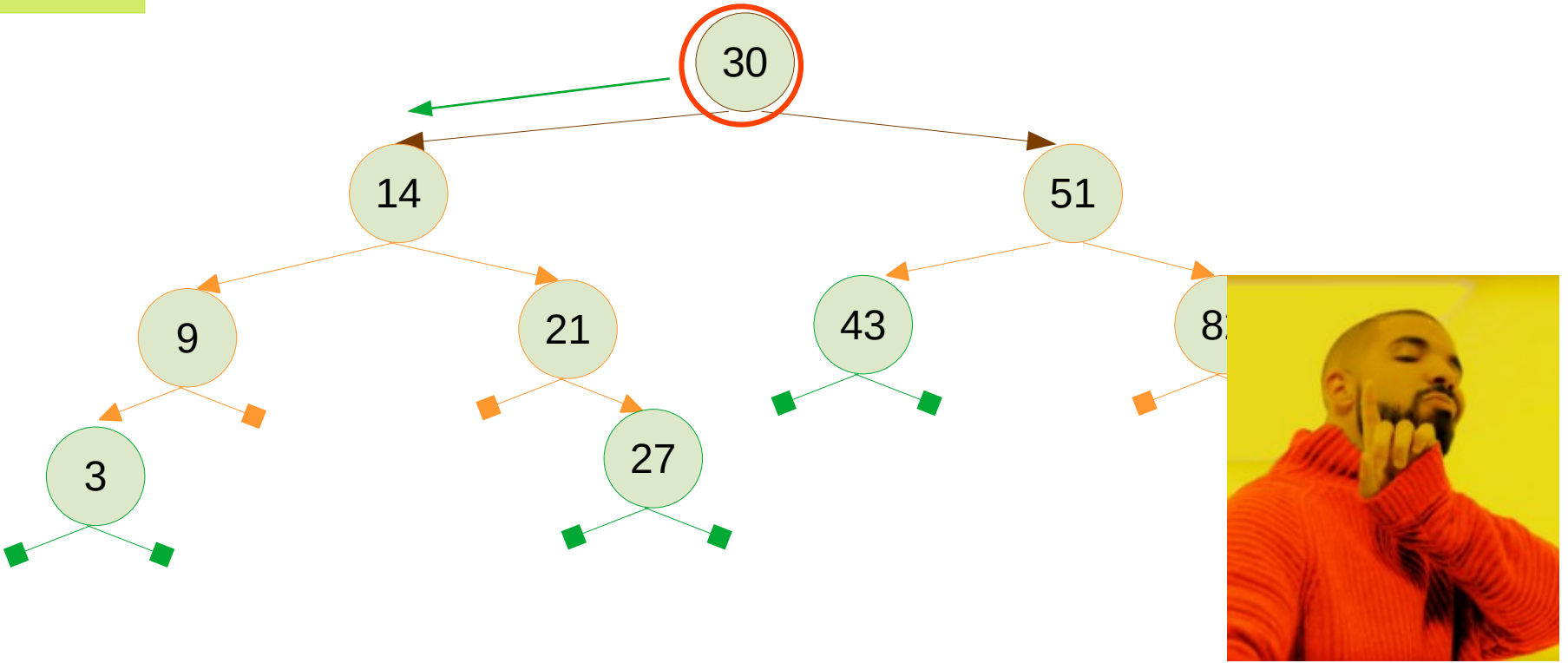
Árvores de busca (Busca)

K = 21



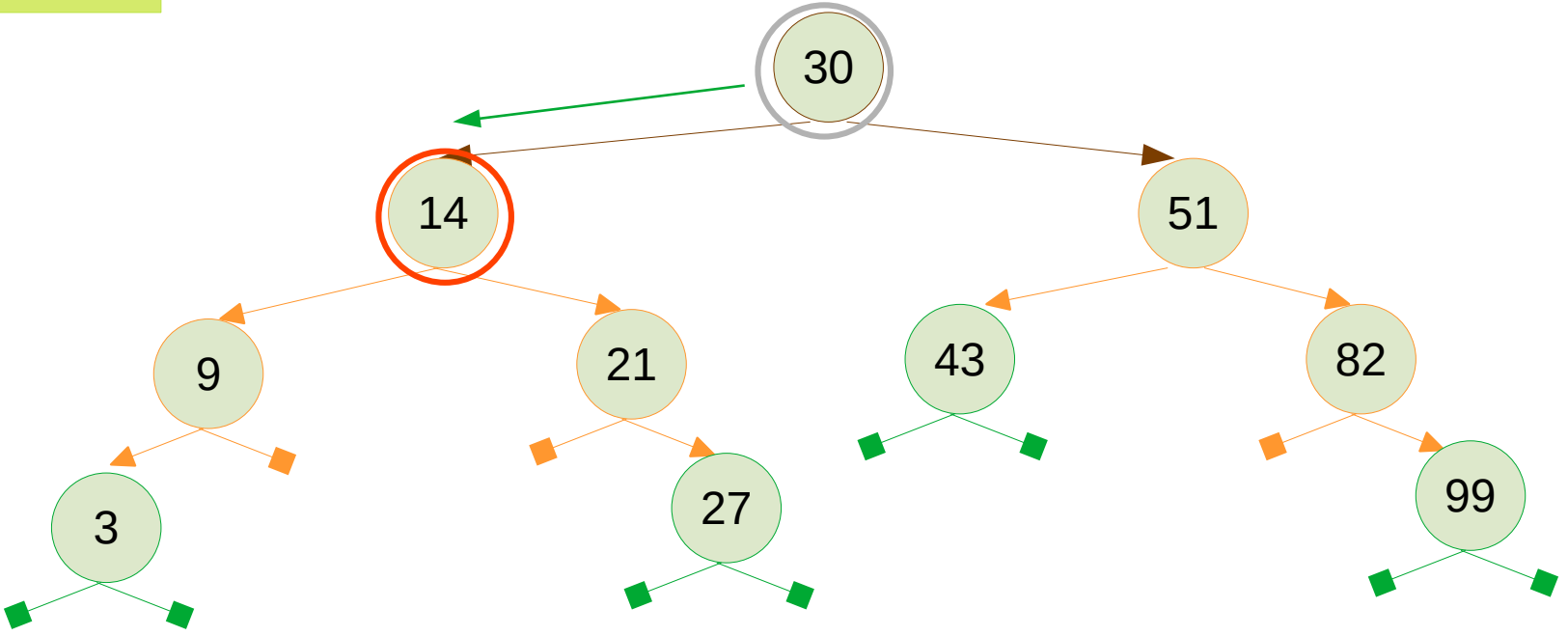
Árvores de busca (Busca)

K = 21



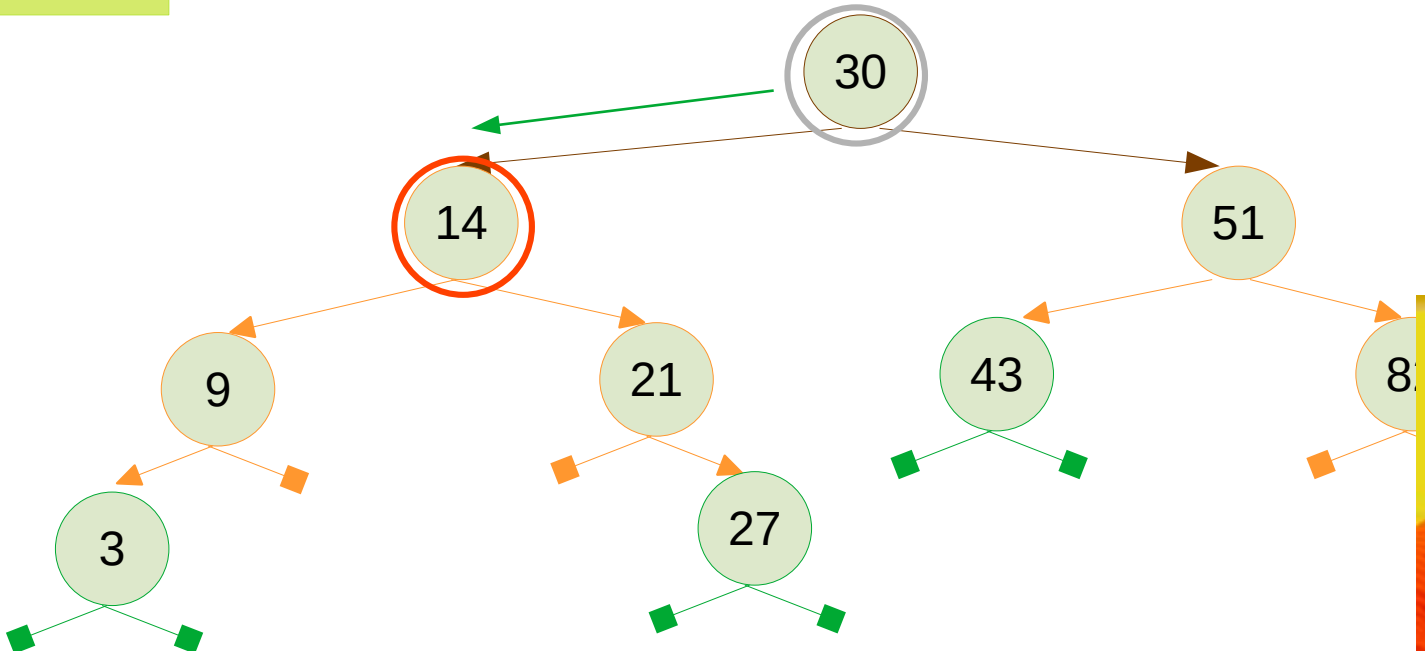
Árvores de busca (Busca)

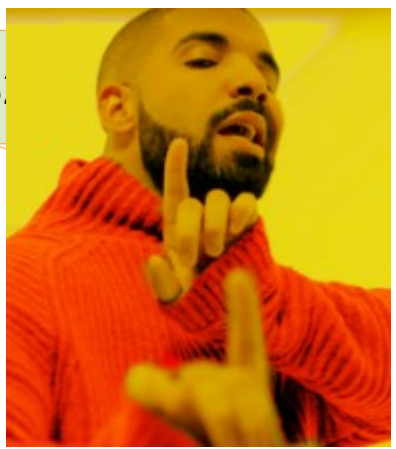
K = 21



Árvores de busca (Busca)

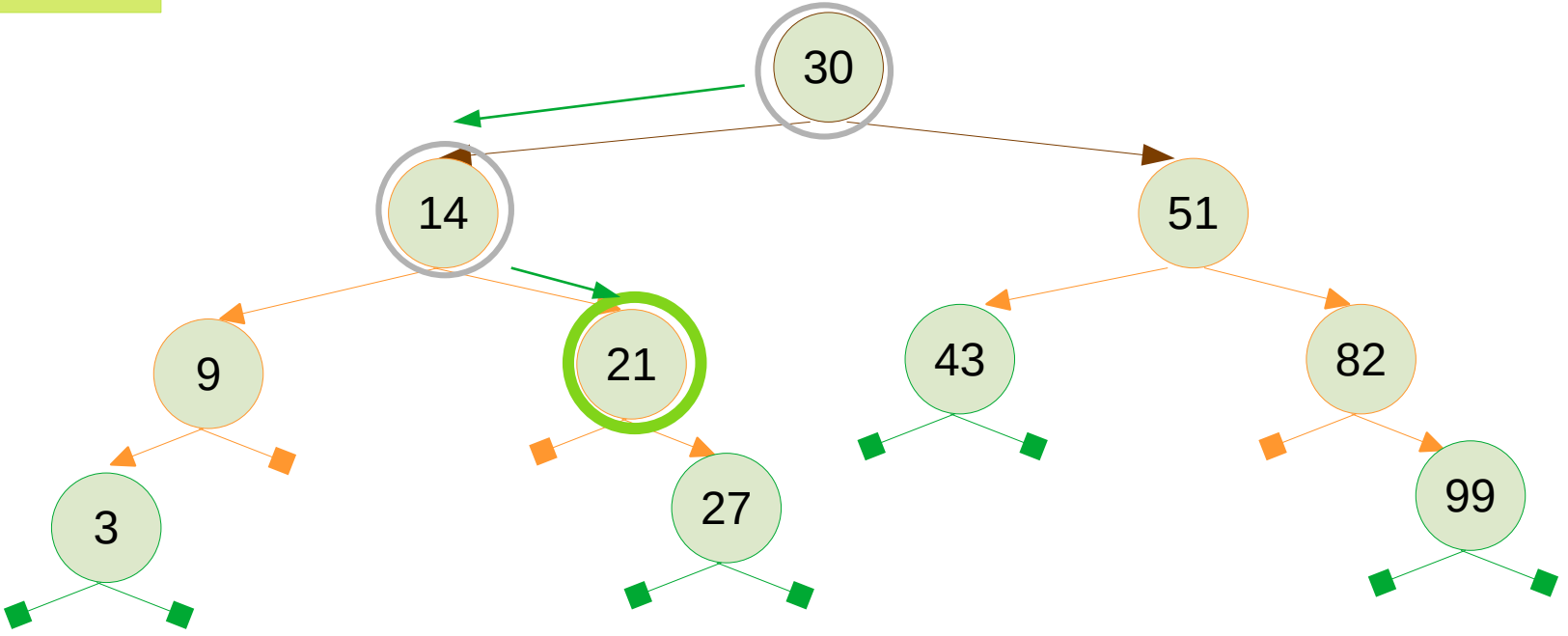
K = 21





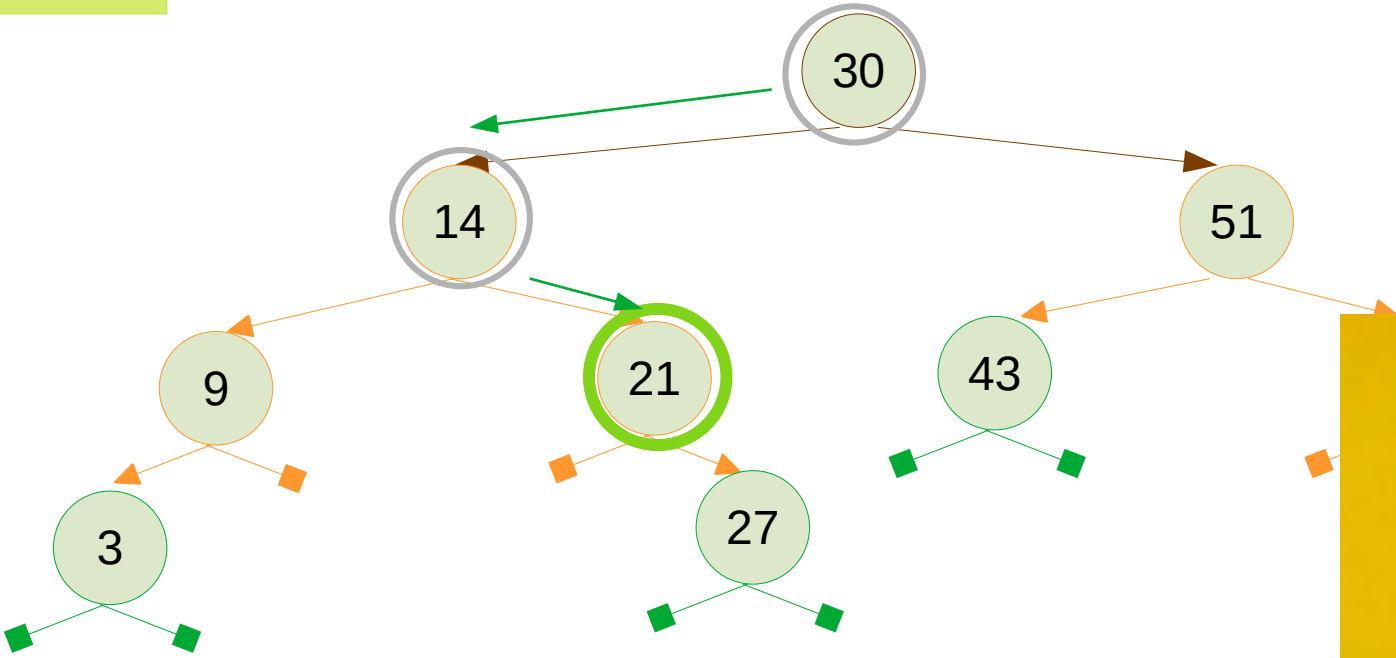
Árvores de busca (Busca)

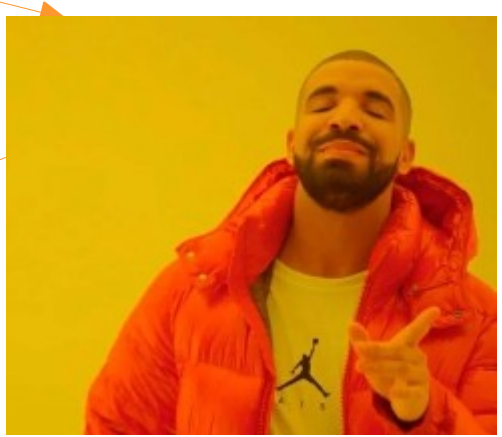
K = 21



Árvores de busca (Busca)

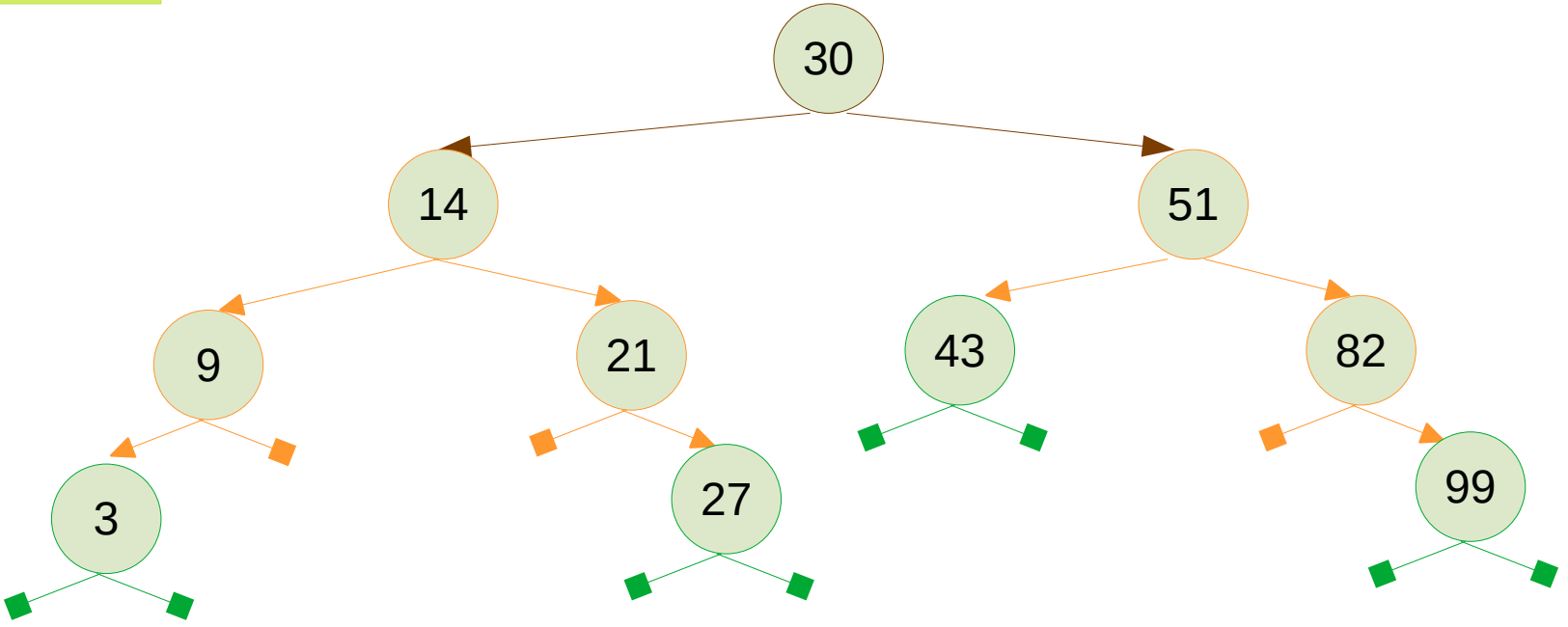
K = 21





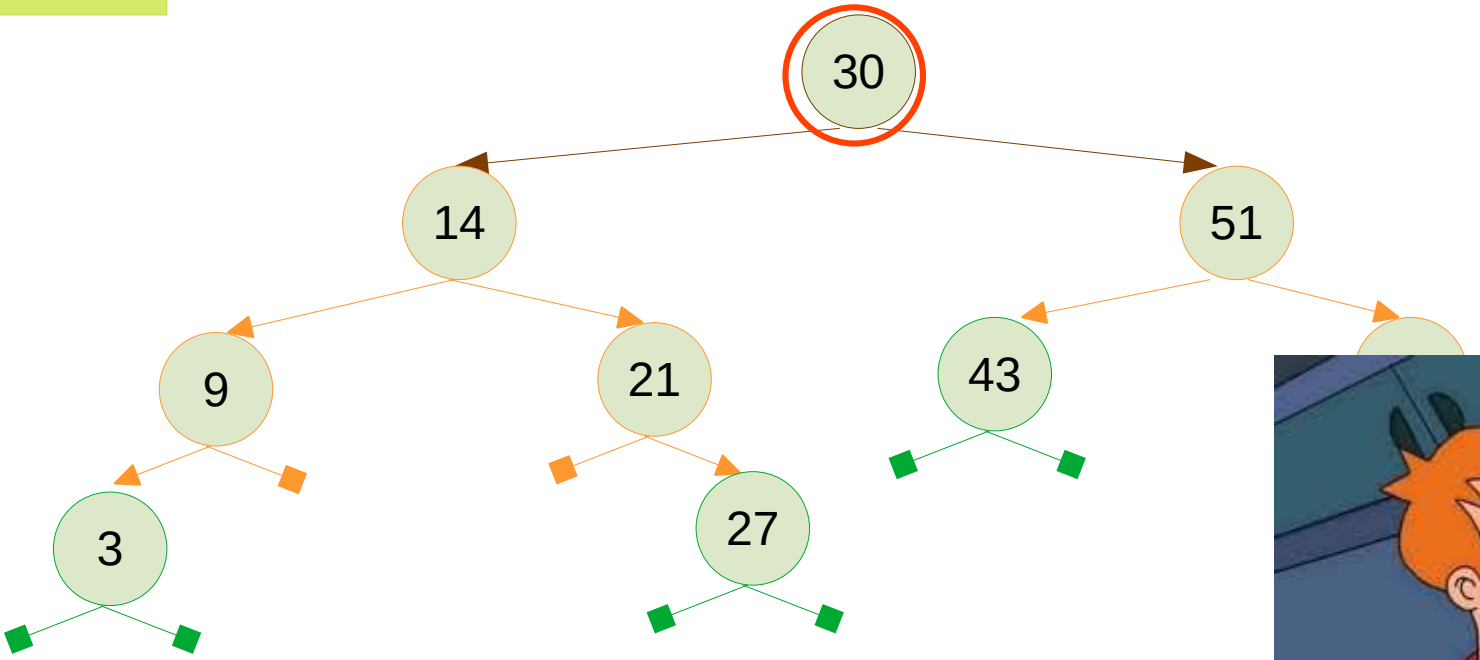
Árvores de busca
(Busca 2)

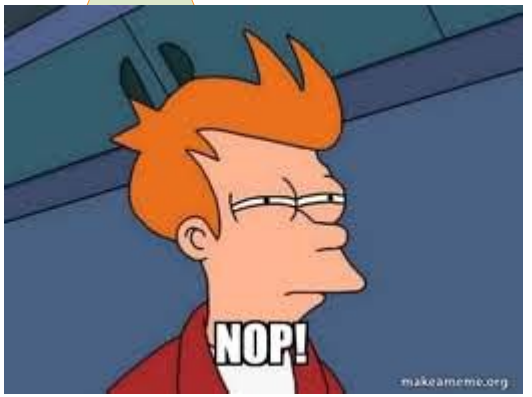
K = 96



Árvores de busca
(Busca 2)

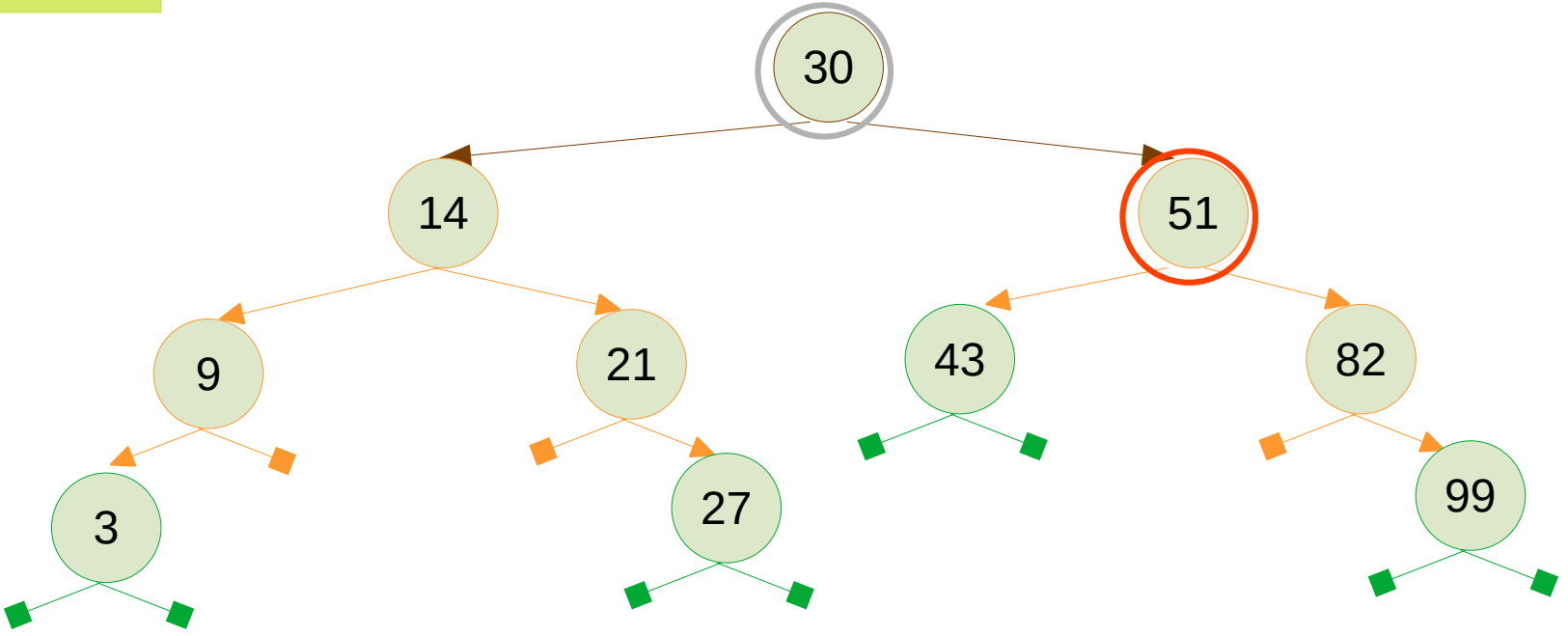
K = 96





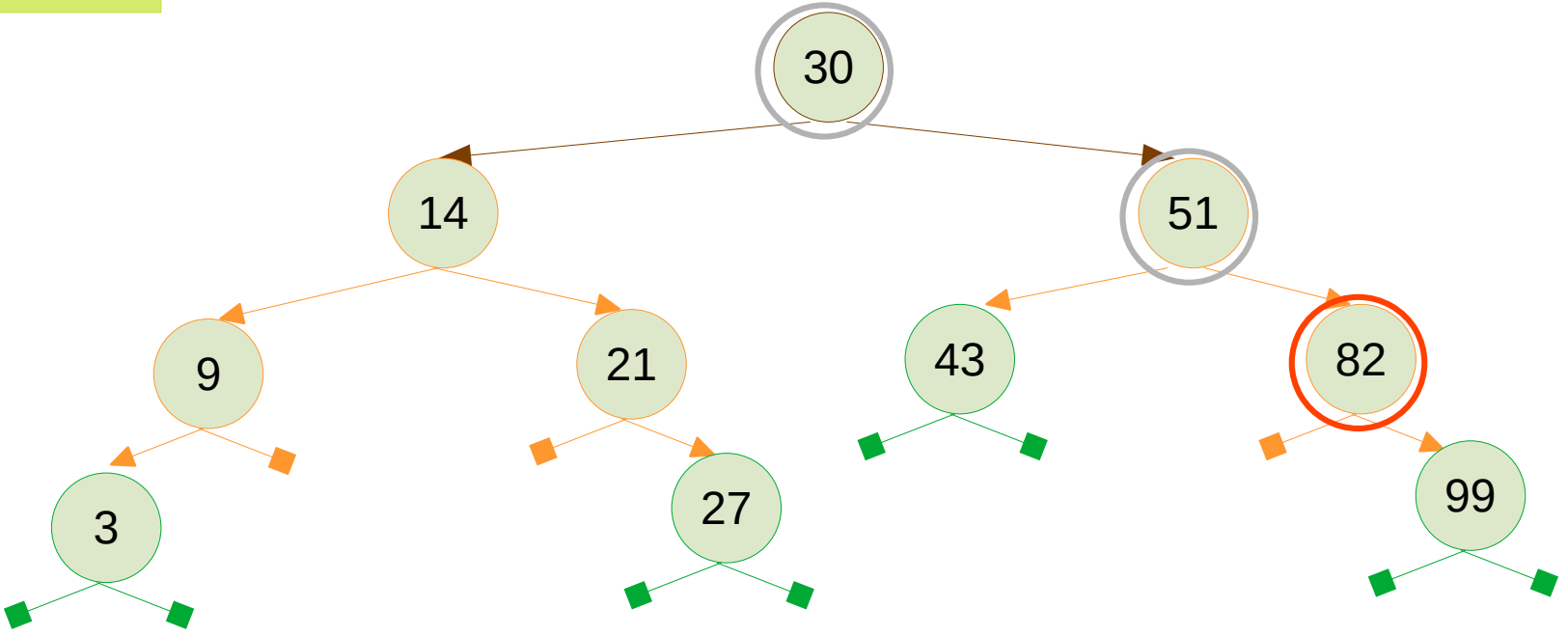
Árvores de busca
(Busca 2)

K = 96



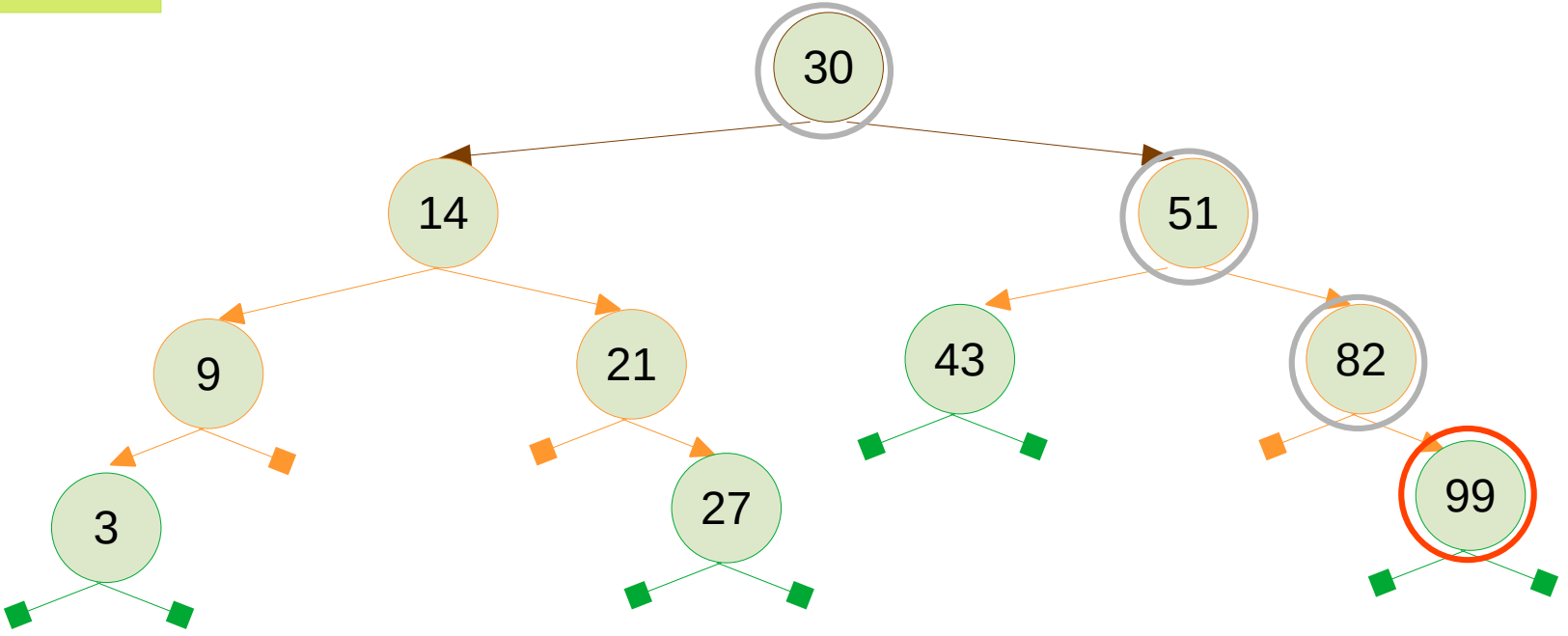
Árvores de busca
(Busca 2)

K = 96



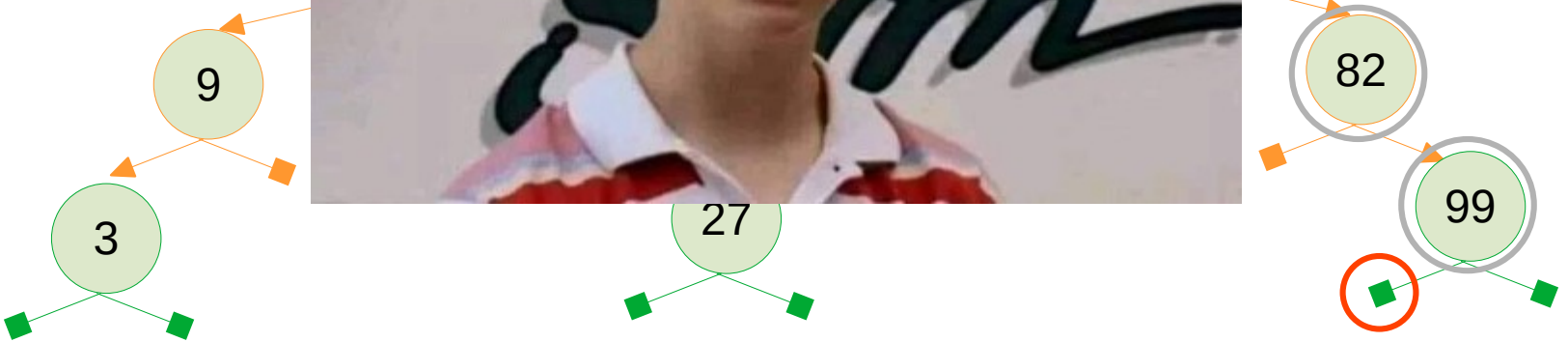
Árvores de busca
(Busca 2)

K = 96



Árvores de busca
(Busca 2)

K = 96



Árvores de busca

Algoritmo buscaChave(Node *raiz, key)

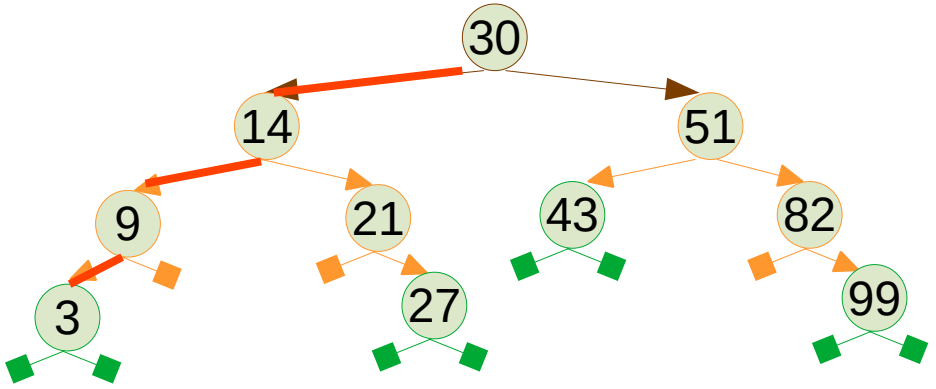
Inicio

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

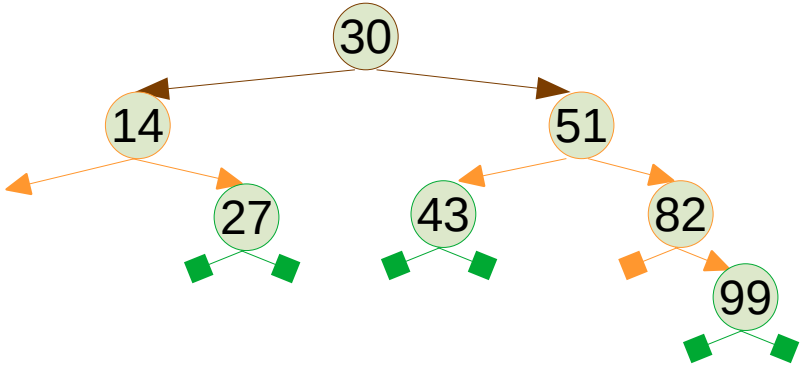
Altura = 4



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.

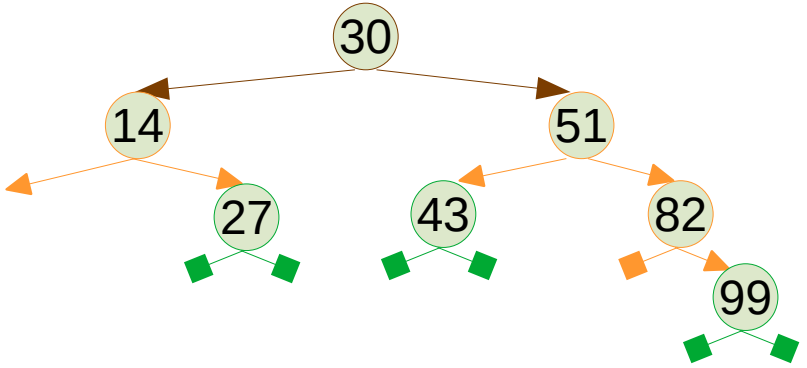
Altura = ?



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

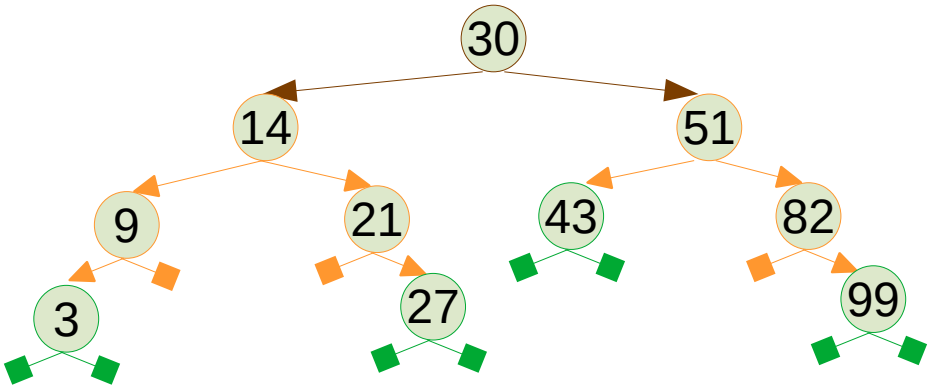
- Existem diversas aplicações onde devemos percorrer uma árvore de maneira sistemática, visitando todos os nós da árvore, um a um.
- Existem três formas de percorrer uma árvore binária:
  - Pré-order
  - In-Order
  - Pós-Order

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

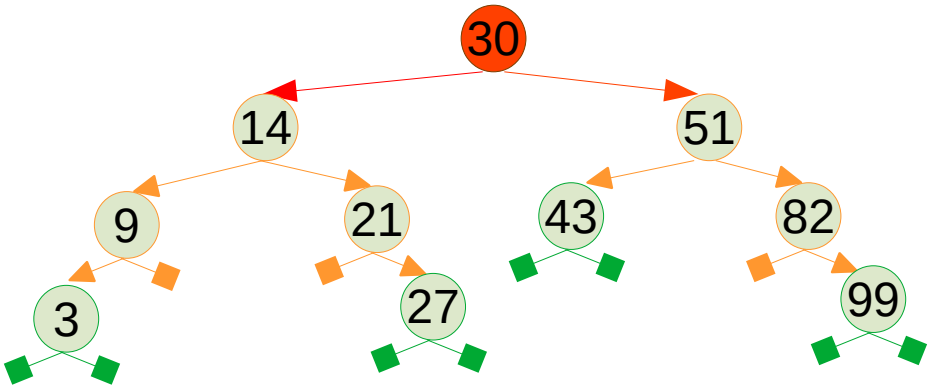
Árvores de busca

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



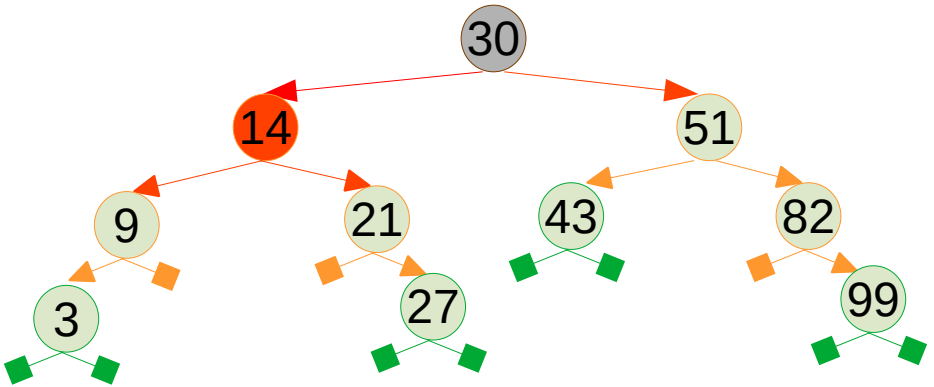
Árvores de busca

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



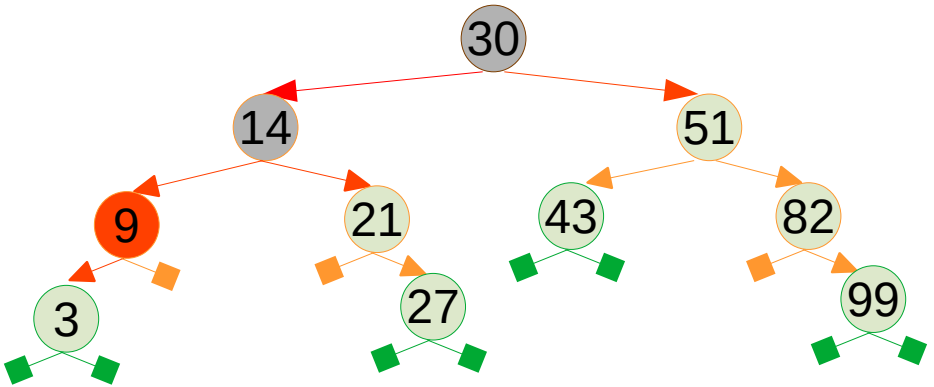
Árvores de busca

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



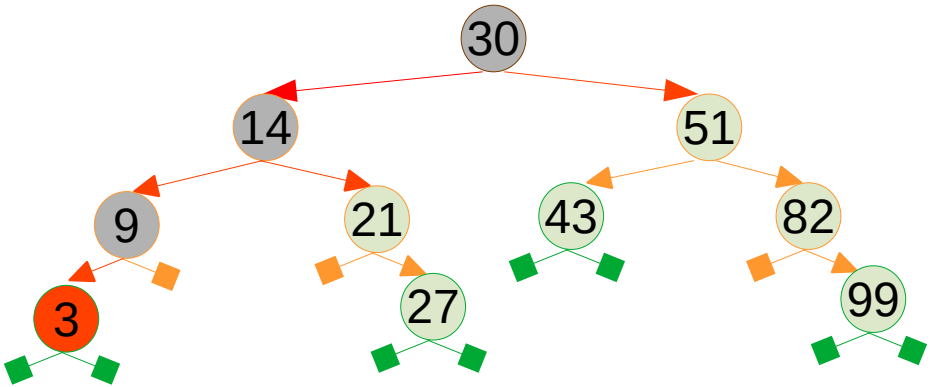
Árvores de busca

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



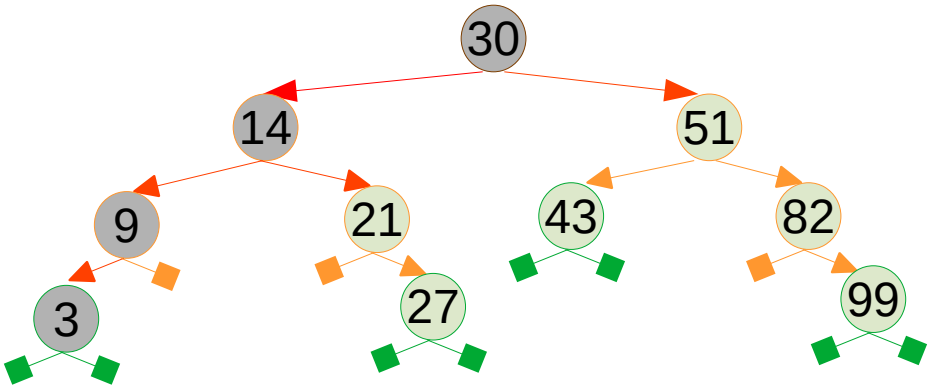
Árvores de busca

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



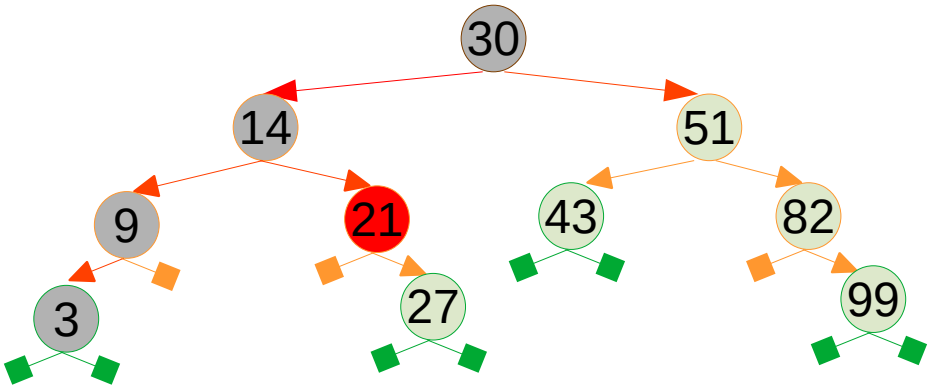
Árvores de busca

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



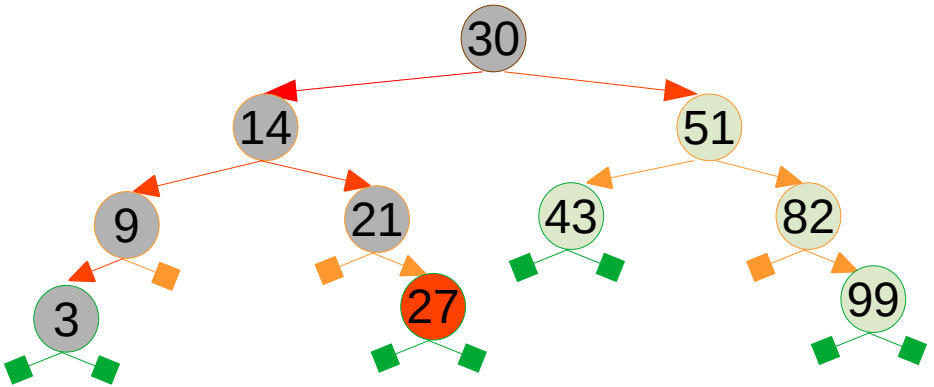
Árvores de busca

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



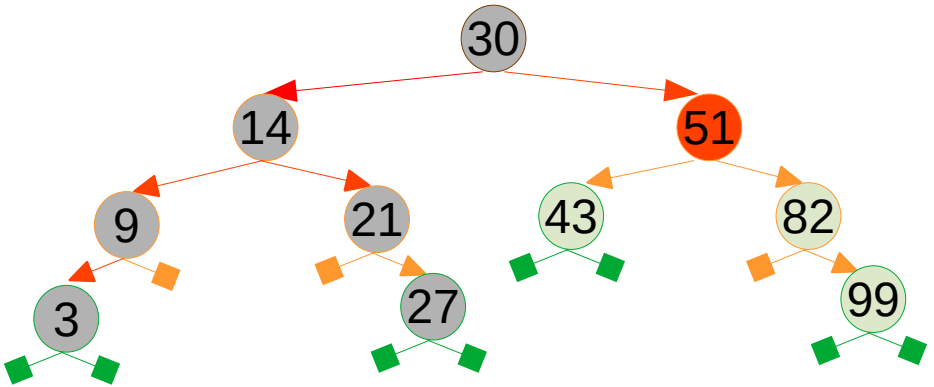
Árvores de busca

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



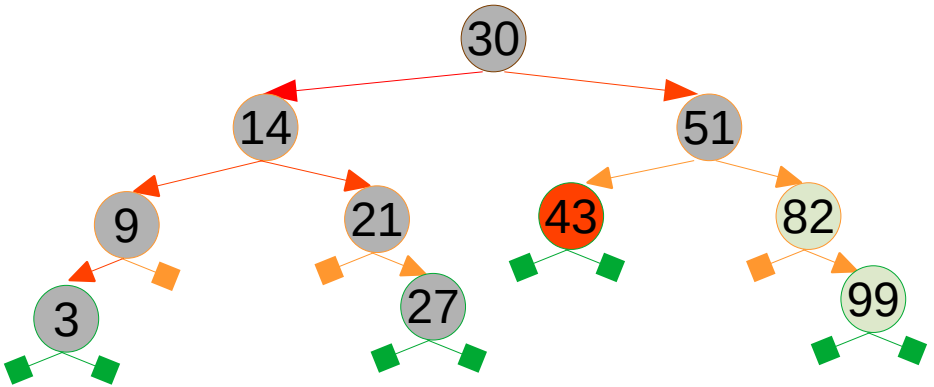
Árvores de busca

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



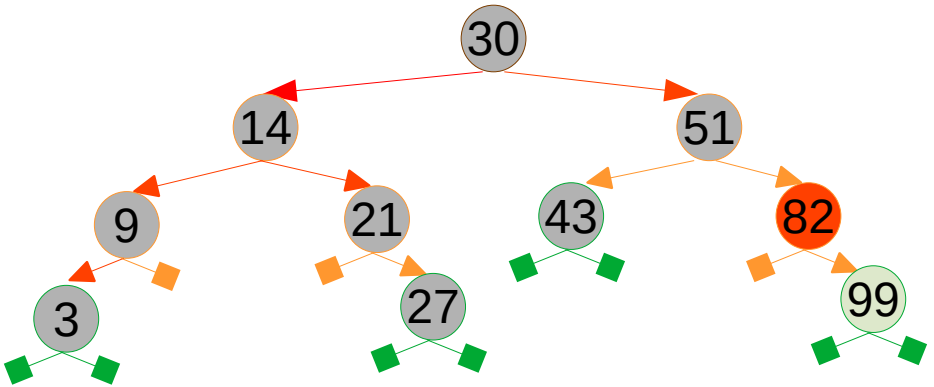
Árvores de busca

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



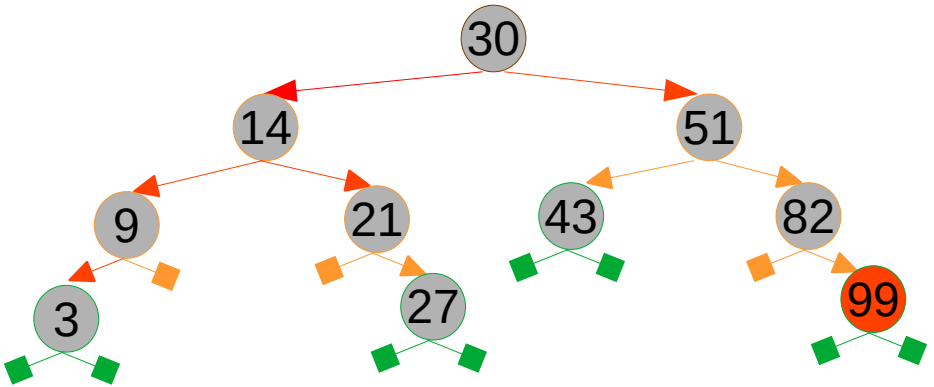
Árvores de busca

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



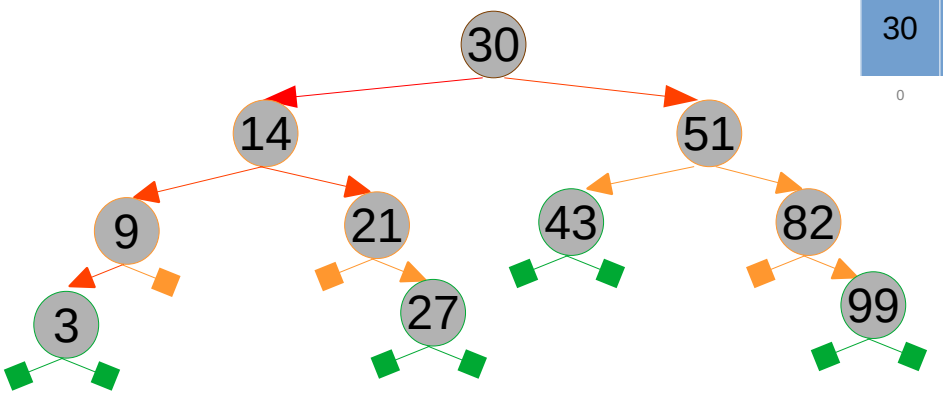
Árvores de busca

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



Árvores de busca

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



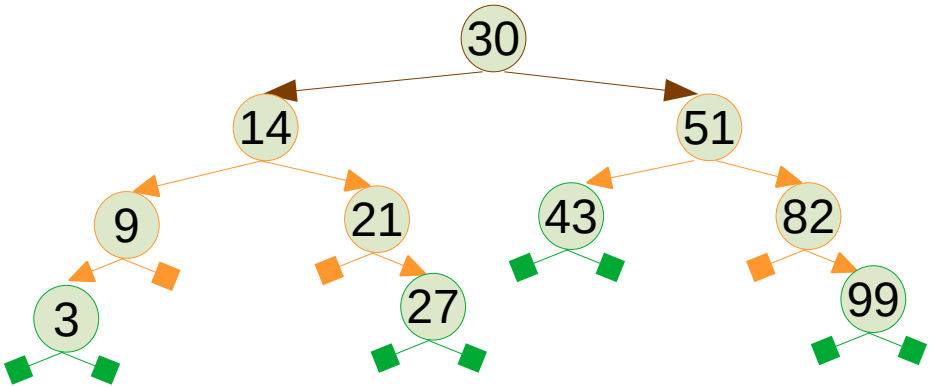
30	14	9	3	21	27	51	43	82	99
0	1	2	3	4	5	6	7	8	9

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

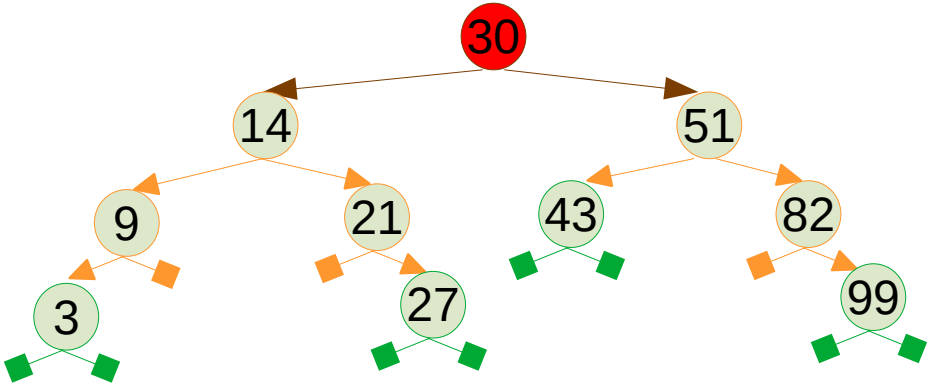
Árvores de busca

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



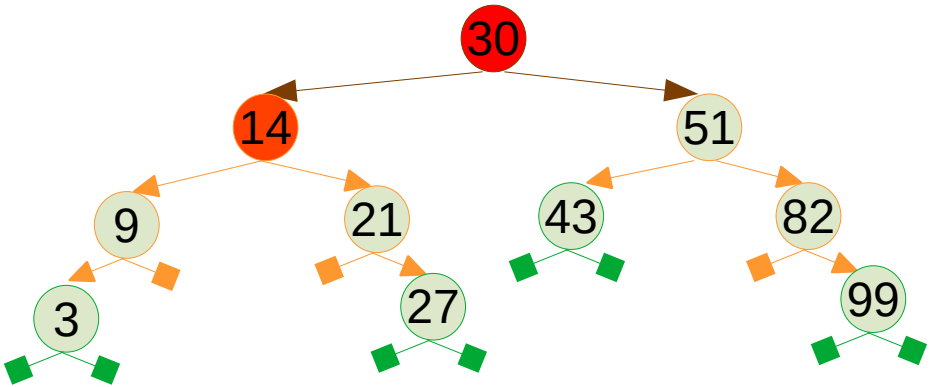
Árvores de busca

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



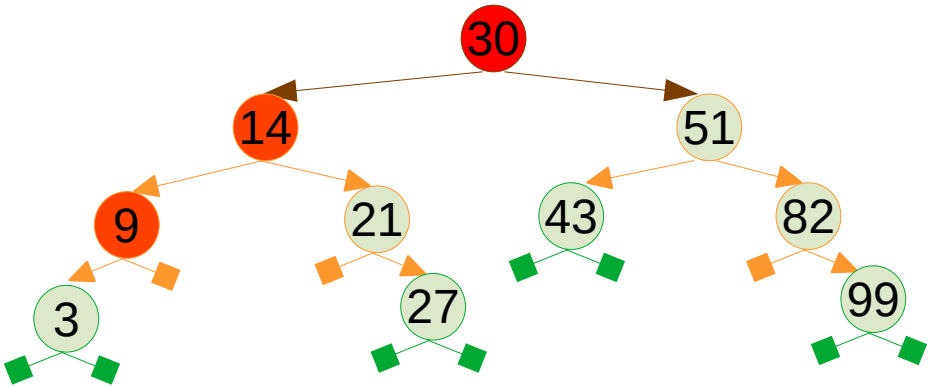
Árvores de busca

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



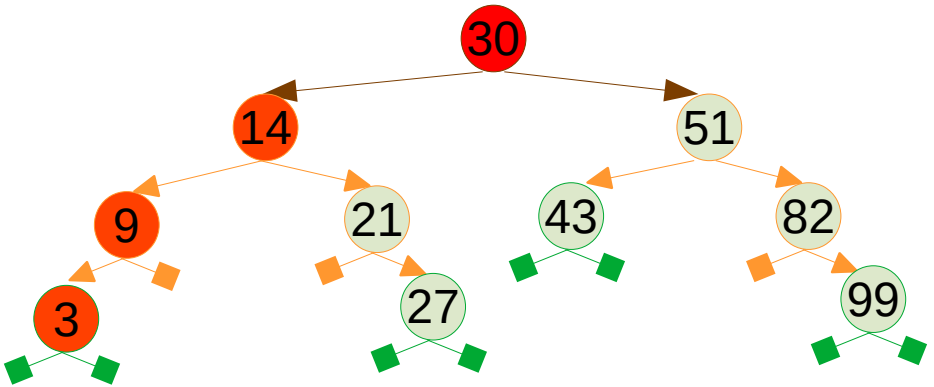
Árvores de busca

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



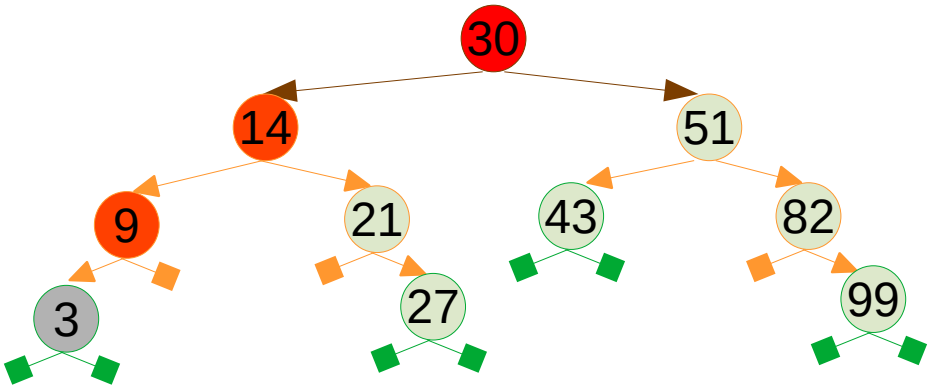
Árvores de busca

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



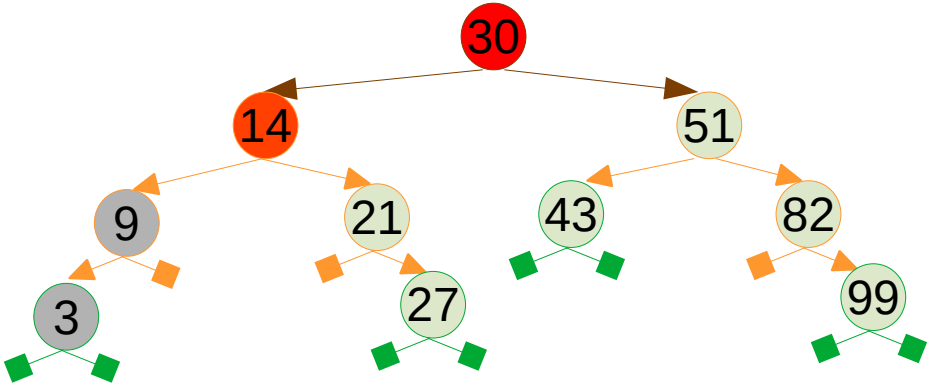
Árvores de busca

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



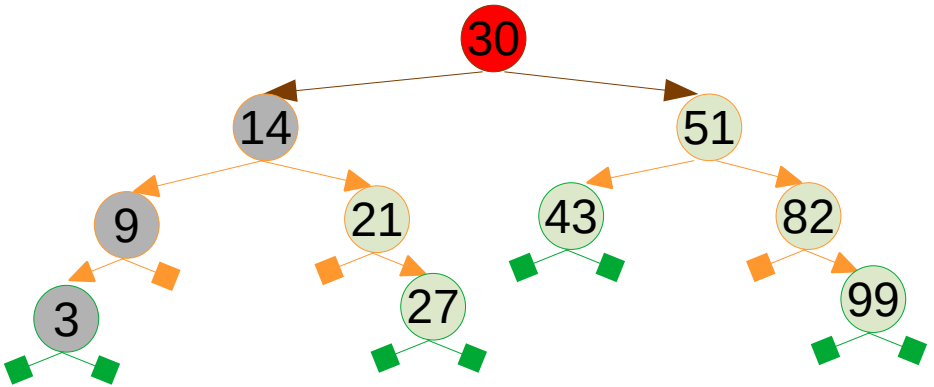
Árvores de busca

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



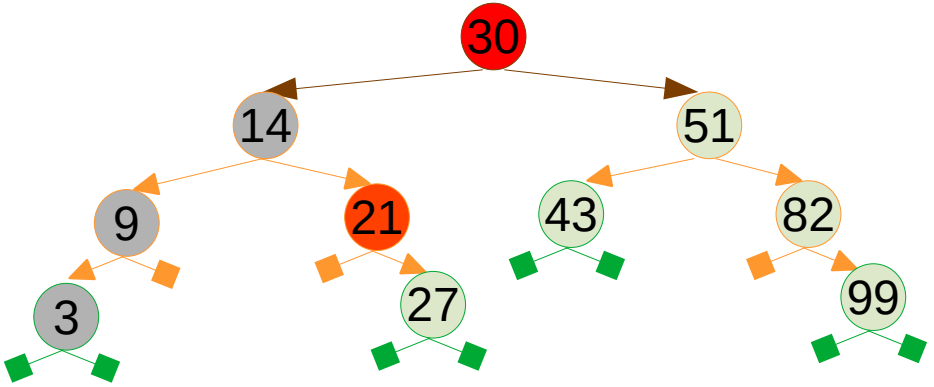
Árvores de busca

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



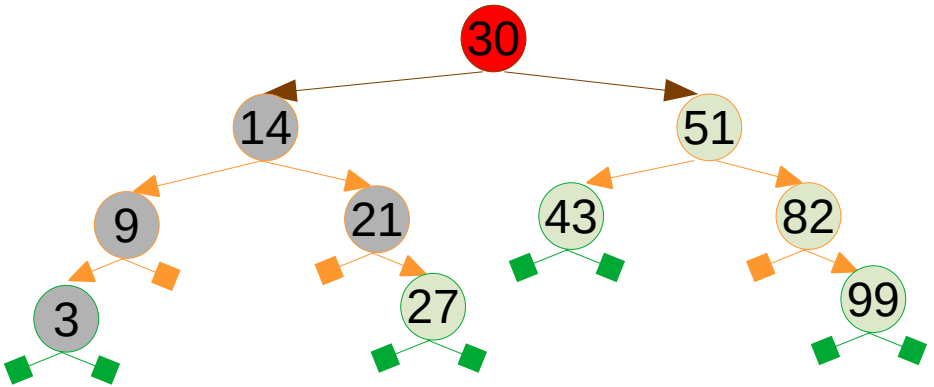
Árvores de busca

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



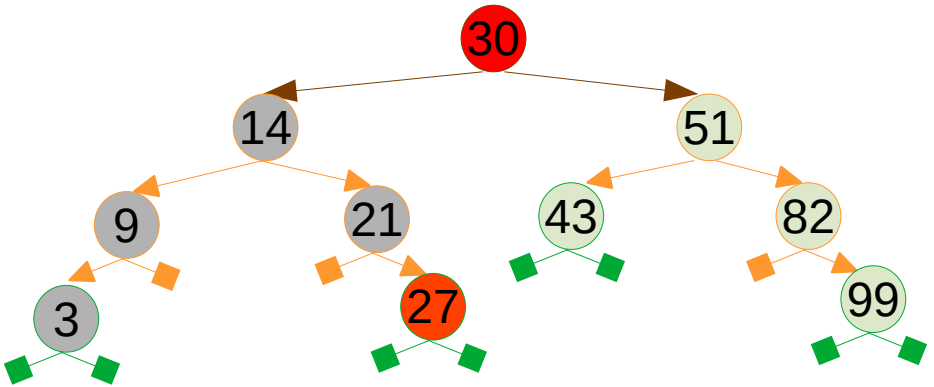
Árvores de busca

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



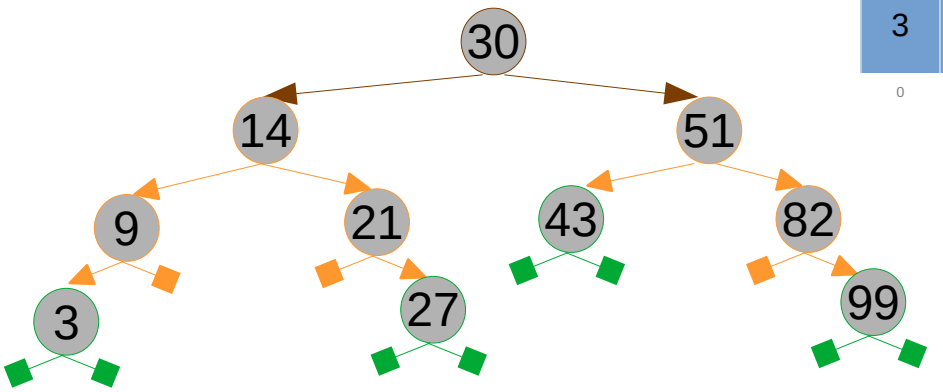
Árvores de busca

- 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



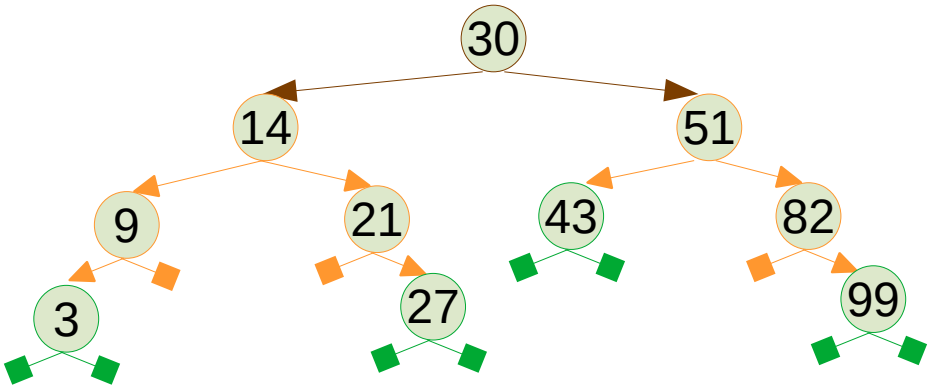
3	9	14	21	27	30	43	51	82	99
0	1	2	3	4	5	6	7	8	9

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**

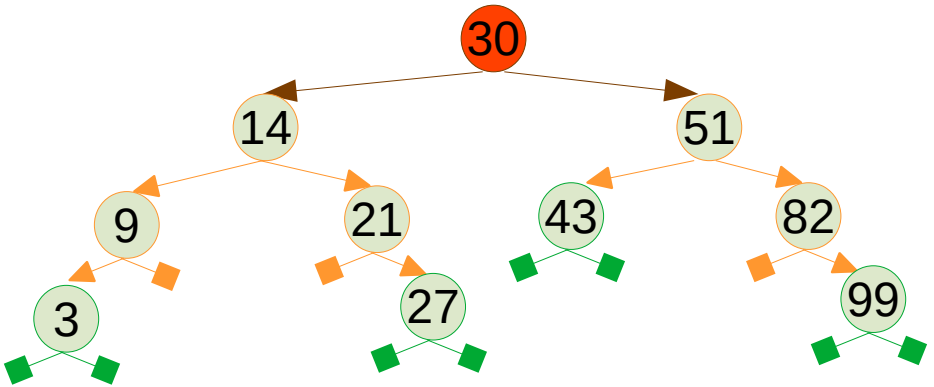
Árvores de busca

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



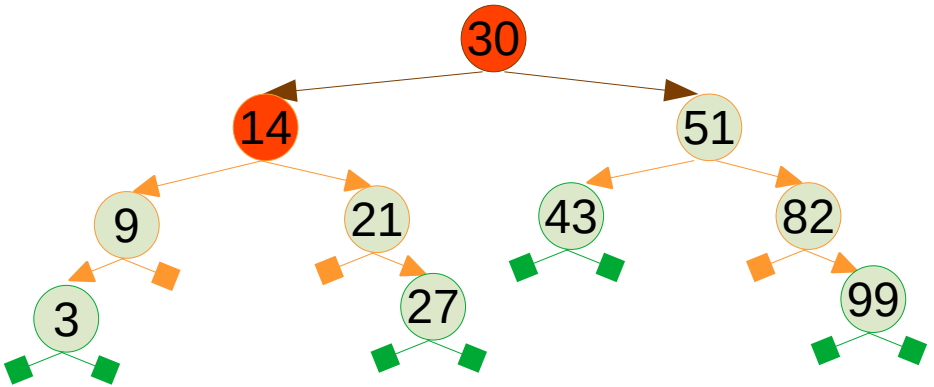
Árvores de busca

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



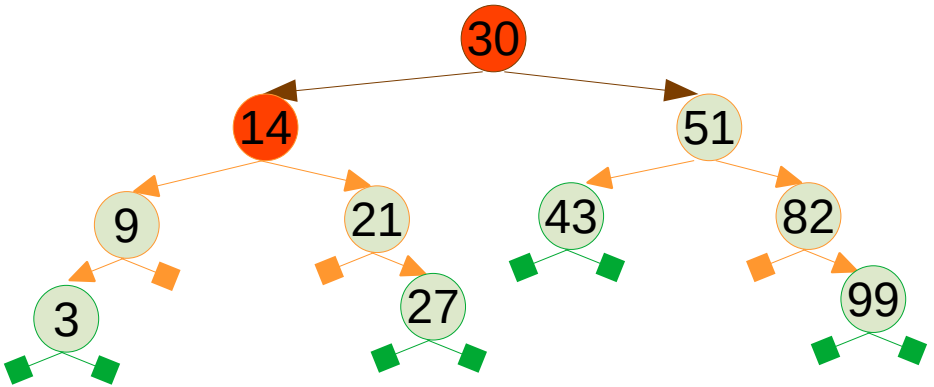
Árvores de busca

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



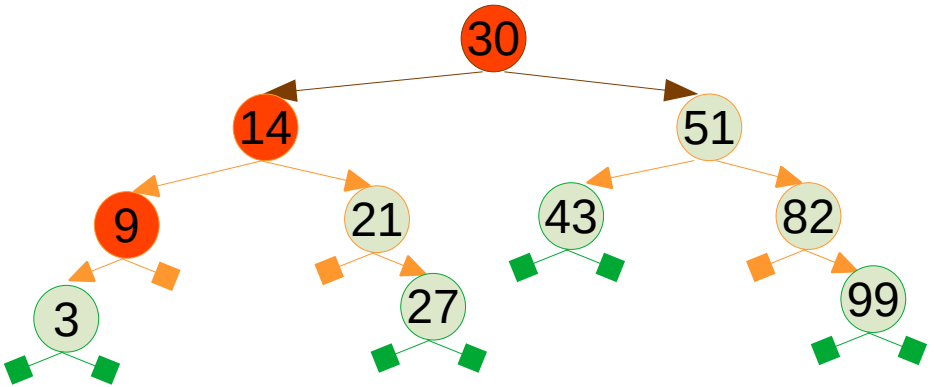
Árvores de busca

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



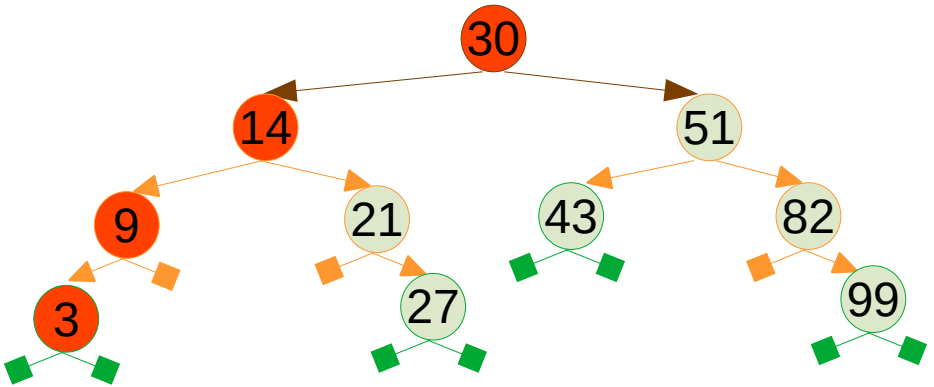
Árvores de busca

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



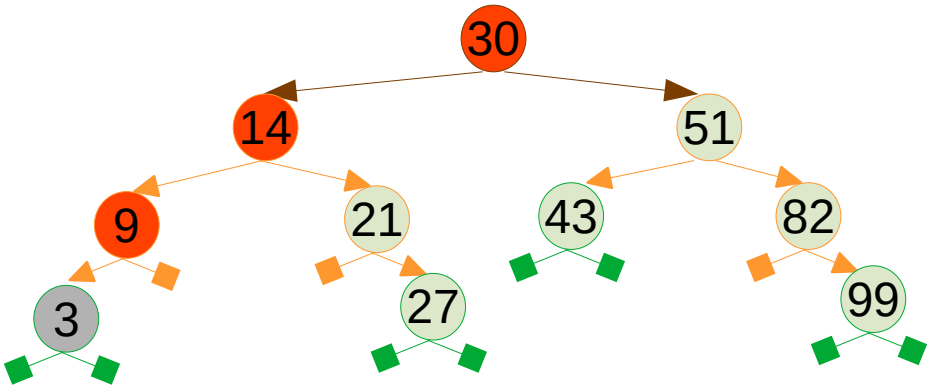
Árvores de busca

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



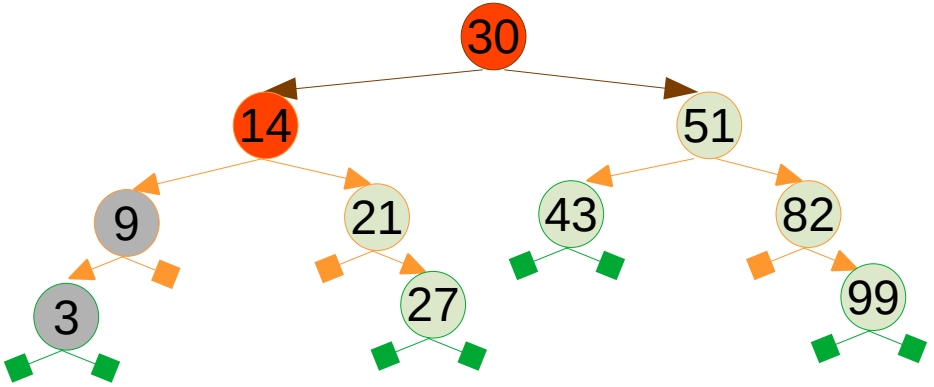
Árvores de busca

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



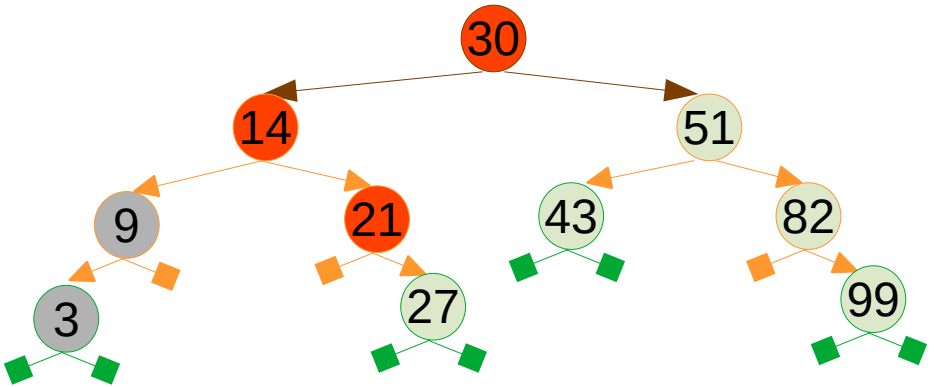
Árvores de busca

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



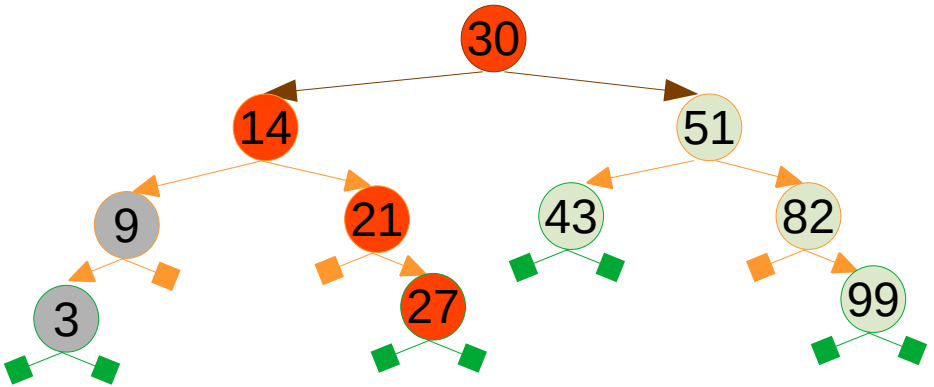
Árvores de busca

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



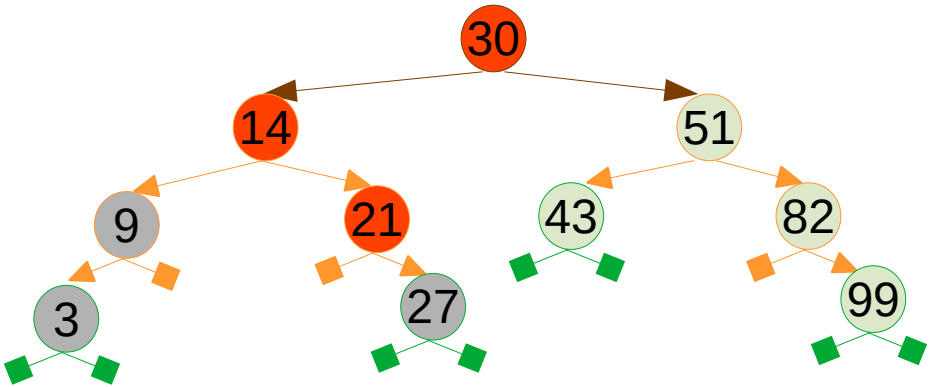
Árvores de busca

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



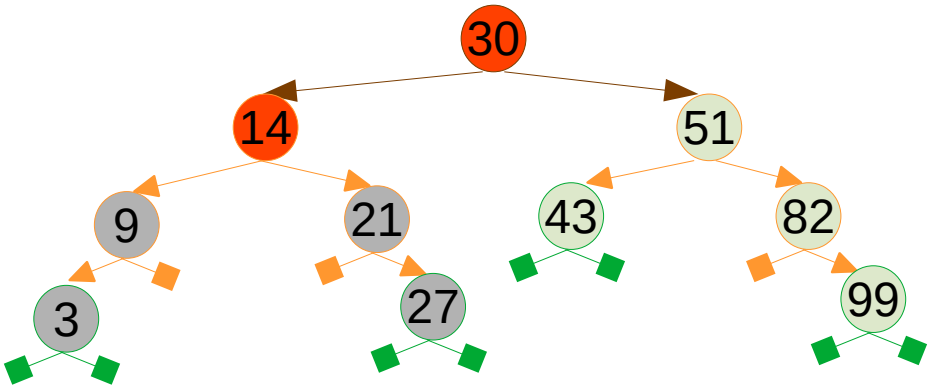
Árvores de busca

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



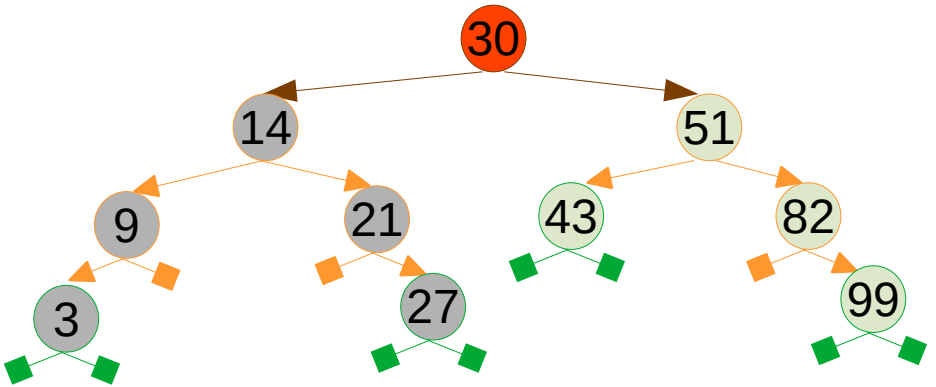
Árvores de busca

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



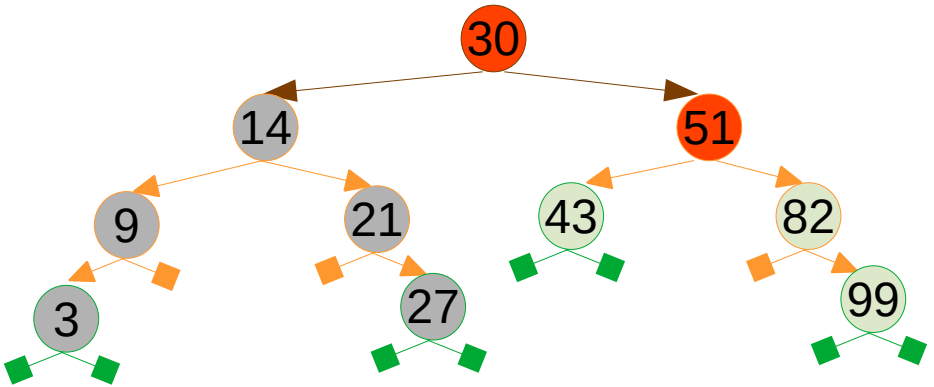
Árvores de busca

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



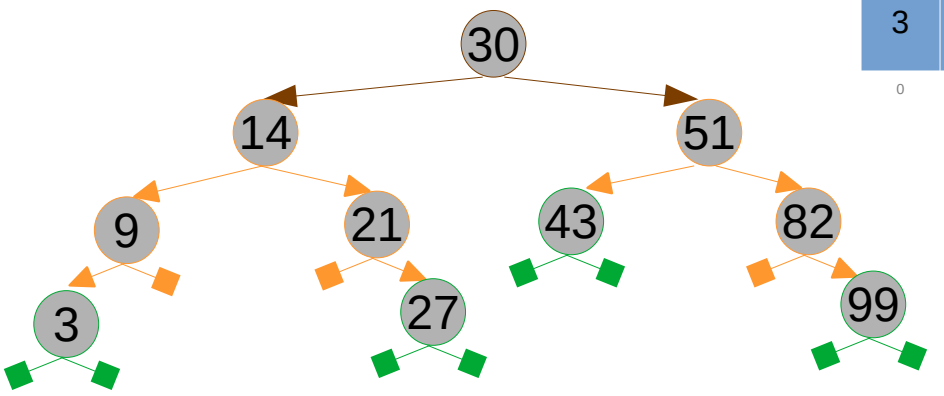
Árvores de busca

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



Árvores de busca

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



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

Árvores de busca

- Árvores representam um ótima otimização de performance para o acesso a informações
  - A complexidade é da ordem de log já que temos uma árvore

Árvores de busca

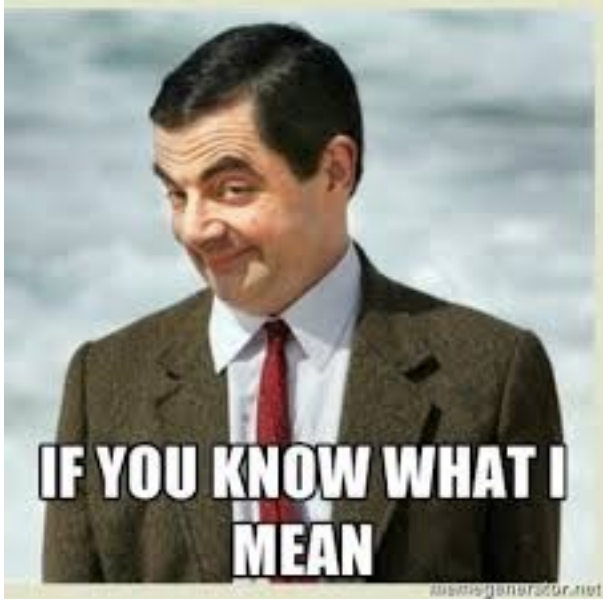
- Árvores representam um ótima otimização de performance para o acesso a informações
  - A complexidade é da ordem de log já que temos uma árvore

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

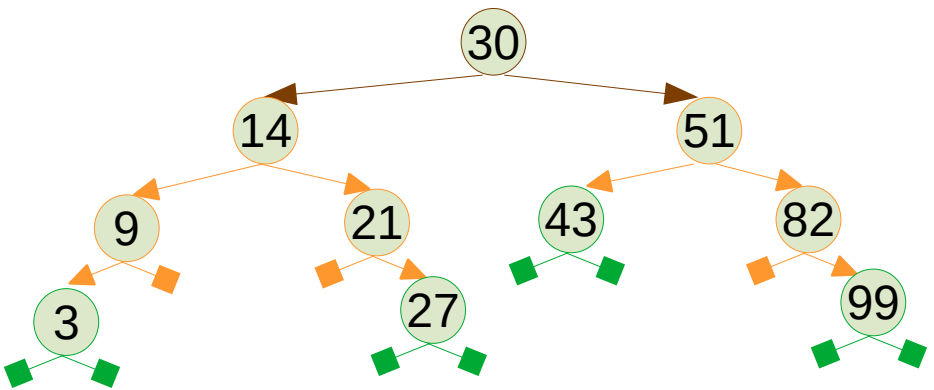
- Árvores degeneradas



Árvores de busca

- Árvores degeneradas

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



Árvores de busca

- Árvores degeneradas

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