

AULA 16

ESTRUTURA DE DADOS

Árvores AVL



Árvore Binária de de Busca (ABB)

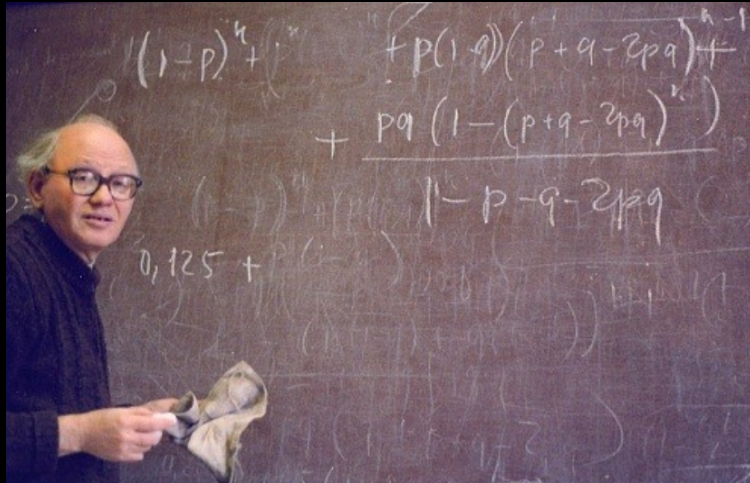
Uma árvore binária é balanceada (ou equilibrada) se, em cada um de seus nós, as subárvores esquerda e direita tem aproximadamente a mesma altura.

n	lg(n)
2	1
32	5
512	9
8192	13
131072	17
2097152	21
33554432	25
536870912	29
8589934592	33
137438953472	37
2199023255552	41
35184372088832	45
562949953421312	49
9007199254740990	53
144115188075856000	57
2305843009213690000	61
36893488147419100000	65
5.9029581035871E+020	69
9.4447329657393E+021	73
1.5111572745183E+023	77
2.4178516392293E+024	81
3.8685626227668E+025	85
6.1897001964269E+026	89
9.9035203142831E+027	93
1.5845632502853E+029	97
2.5353012004565E+030	101
4.0564819207303E+031	105
6.4903710731685E+032	109
1.0384593717070E+034	113
8.3076749736557E+034	116

Árvore Balanceada

- As ABB permitem buscar de forma eficiente um elemento
 - Deseja-se que o custo de acesso tenha a ordem de grandeza de uma árvore ótima $\rightarrow O(\lg(n))$
 - Este custo deve-se manter ao longo da utilização da estrutura (inclusive após inserções/remoções).
 - O custo de ter a estrutura balanceada deve estar, idealmente, na mesma ordem de grandeza.
-
-

Árvore AV-L



Georgy M. Adelson-Velsky
Russia
(1922-2014/abril/26)



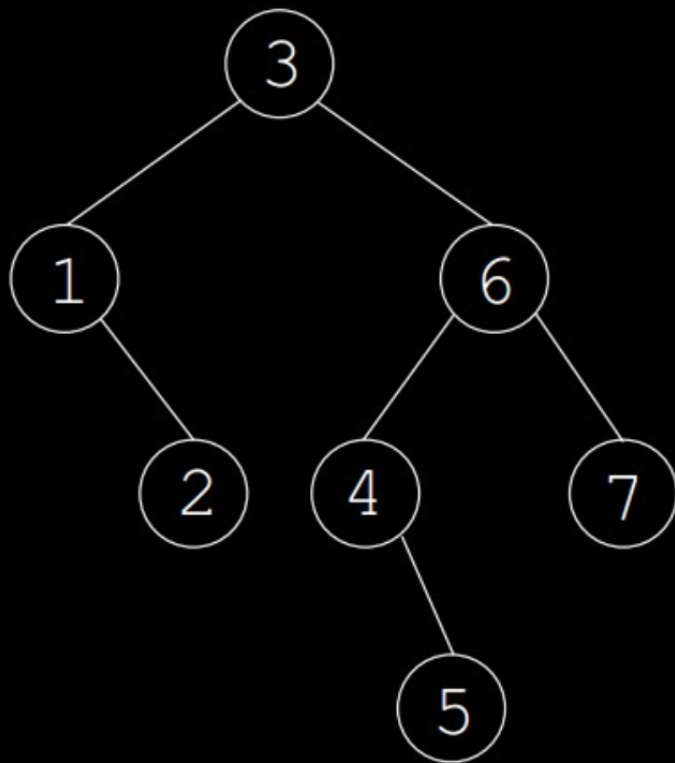
Evgenii Mikhailovich Landis
Ucrania (1921-1997)

Árvore AV-L : Definição

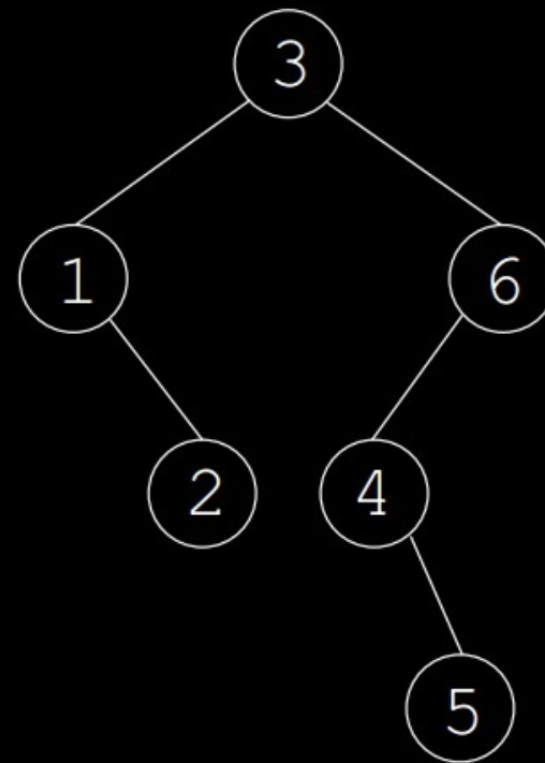
- Altura de um nó: número de passos do mais longo caminho até uma folha
- Para cada nó na árvore, a diferença de altura de suas duas subárvores é no máximo 1 (AVL)



Árvore AV-L



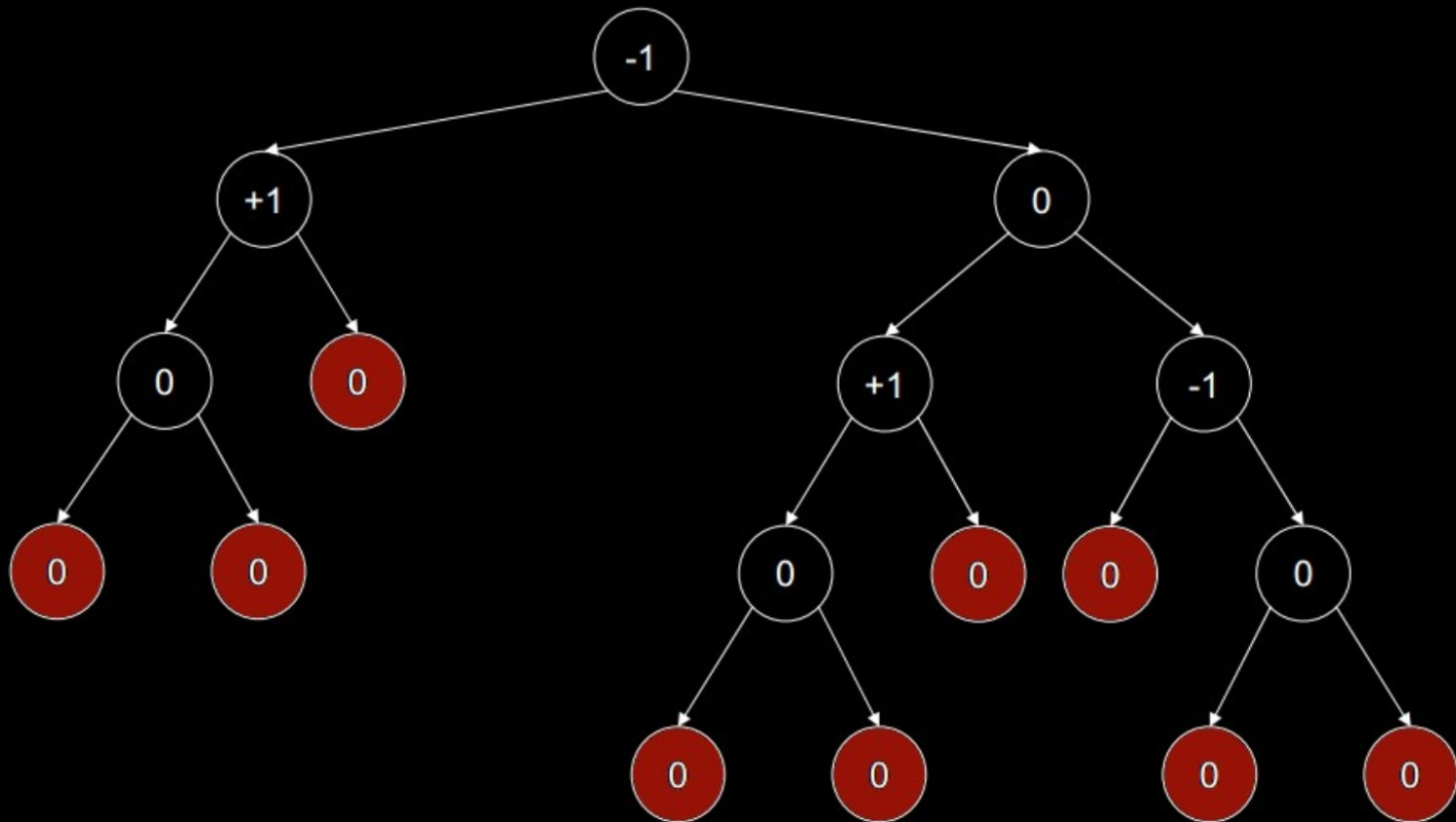
AVL



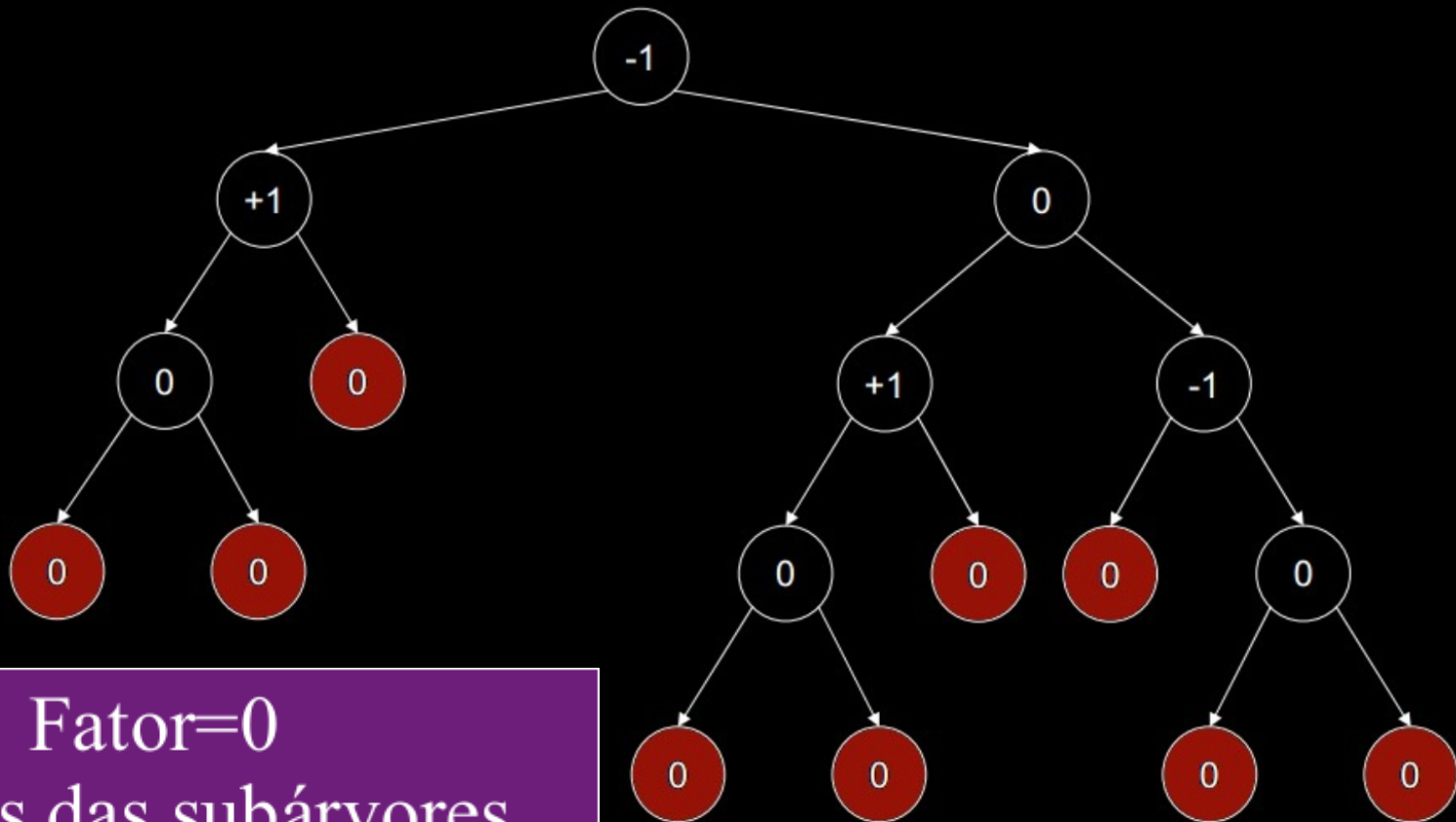
NÃO AVL

Fator de Balanceamento

- O fator de balanceamento/equilíbrio de um nó T em uma ABB é definido como: $h_L - h_R$
- Para qualquer nó T em uma árvore AVL, o fator de balanceamento assume o valor: +1, 0, -1.
- O fator de balanceamento de uma folha?



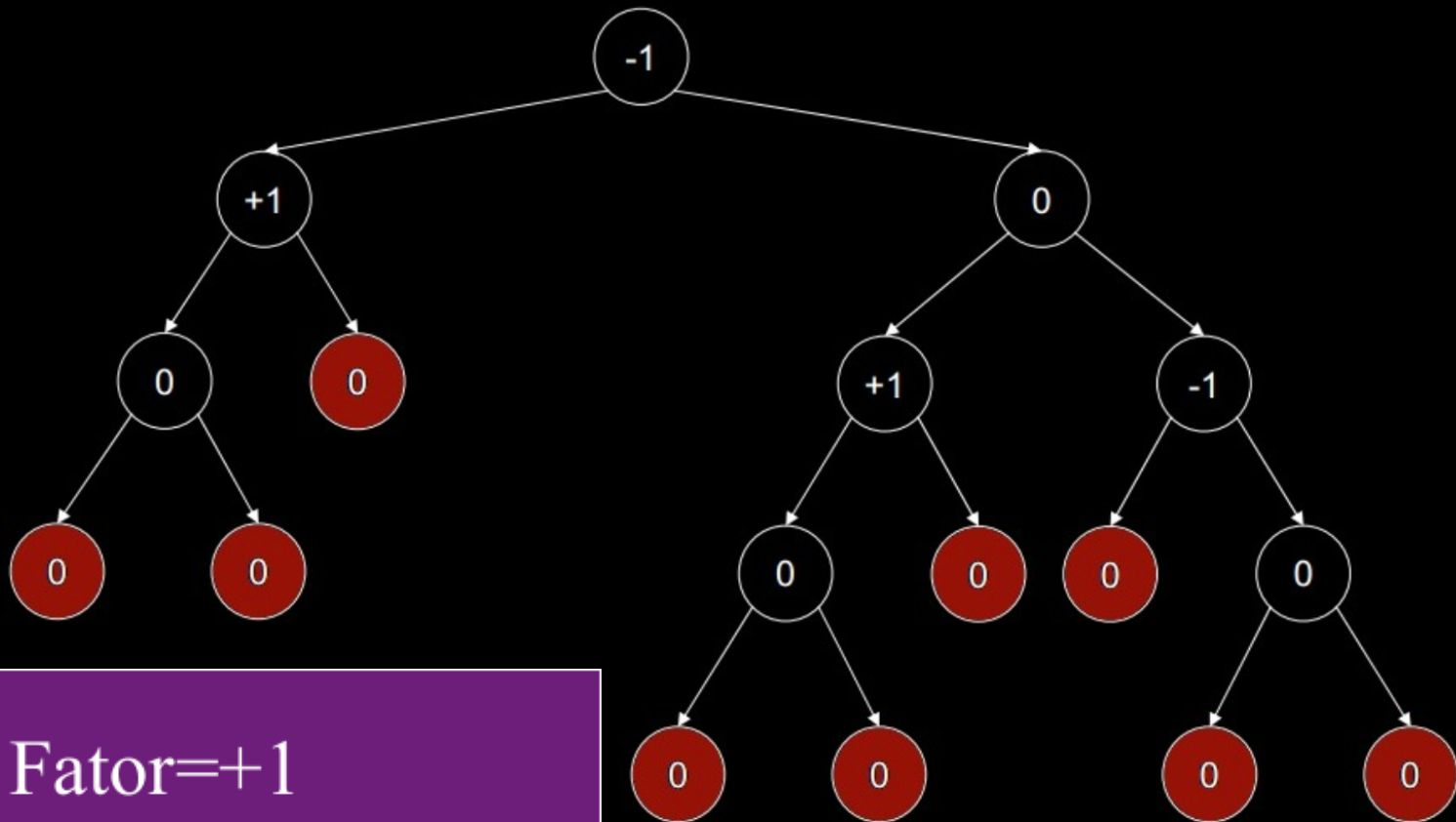
Fator de Balanceamento



Fator=0

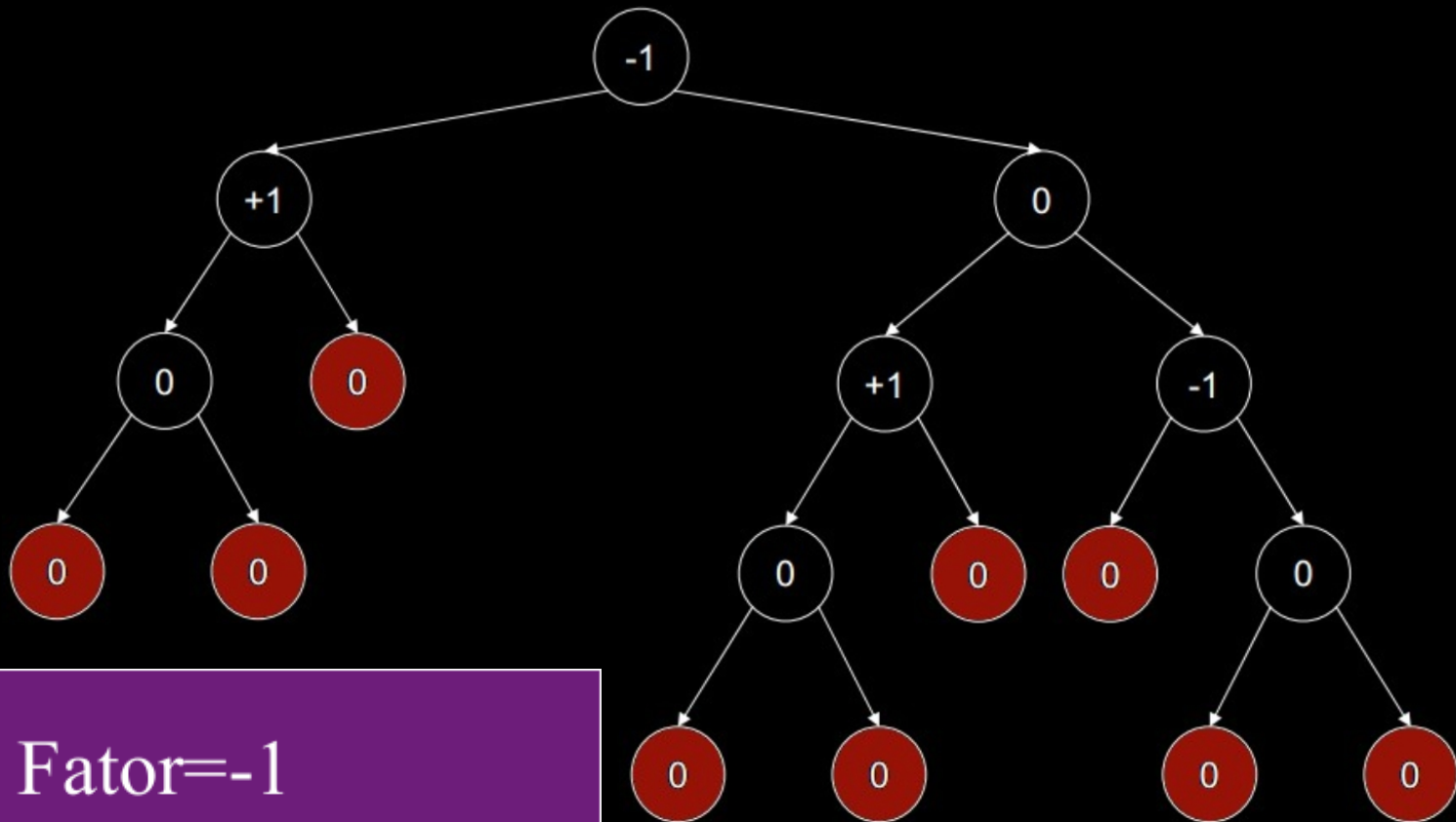
Alturas das subárvores
esquerda e direita
são iguais

Fator de Balanceamento



Fator=+1
Alturas da subárvore
esquerda é maior

Fator de Balanceamento



Fator=-1
Alturas da subárvore
direita é maior

Inserção de elementos

- Maio
 - Março
 - Novembro
 - Agosto
 - Abril
 - Janeiro
 - Dezembro
 - Julho
 - Fevereiro
 - Junho
 - Outubro
 - Setembro
-
-

Inserção de elementos

- Maio
 - Março
 - Novembro
 - Agosto
 - Abril
 - Janeiro
 - Dezembro
 - Julho
 - Fevereiro
 - Junho
 - Outubro
 - Setembro
-
-

Inserção de elementos – Maio

Depois da inserção

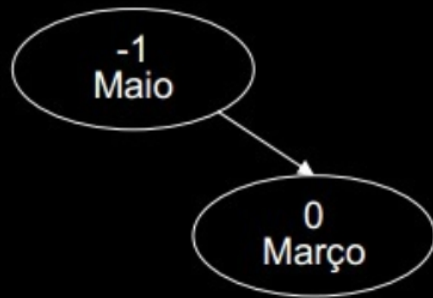


Depois do rebalanceamento

*Sem necessidade
de rebalanceamento*

Inserção de elementos - Março

Depois da inserção

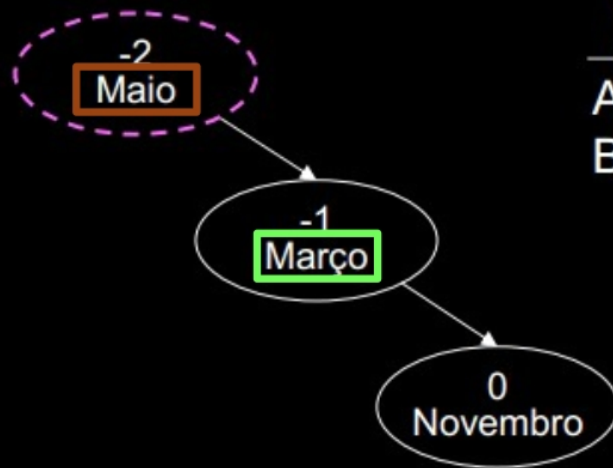


Depois do rebalanceamento

*Sem necessidade
de rebalanceamento*

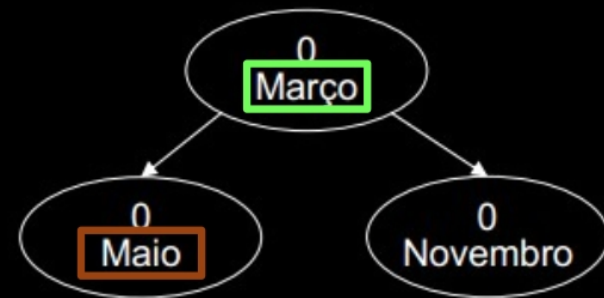
Inserção de elementos - Novembro

Depois da inserção



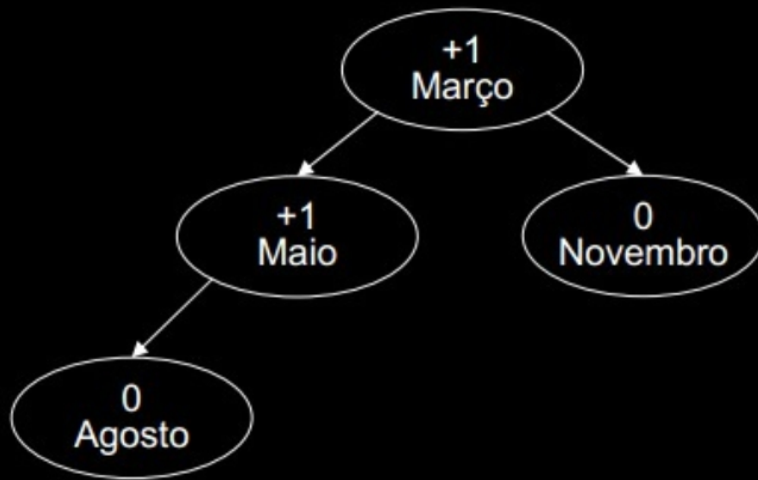
RR
A = -2
B = -1

Depois do rebalanceamento



Inserção de elementos - Agosto

Depois da inserção



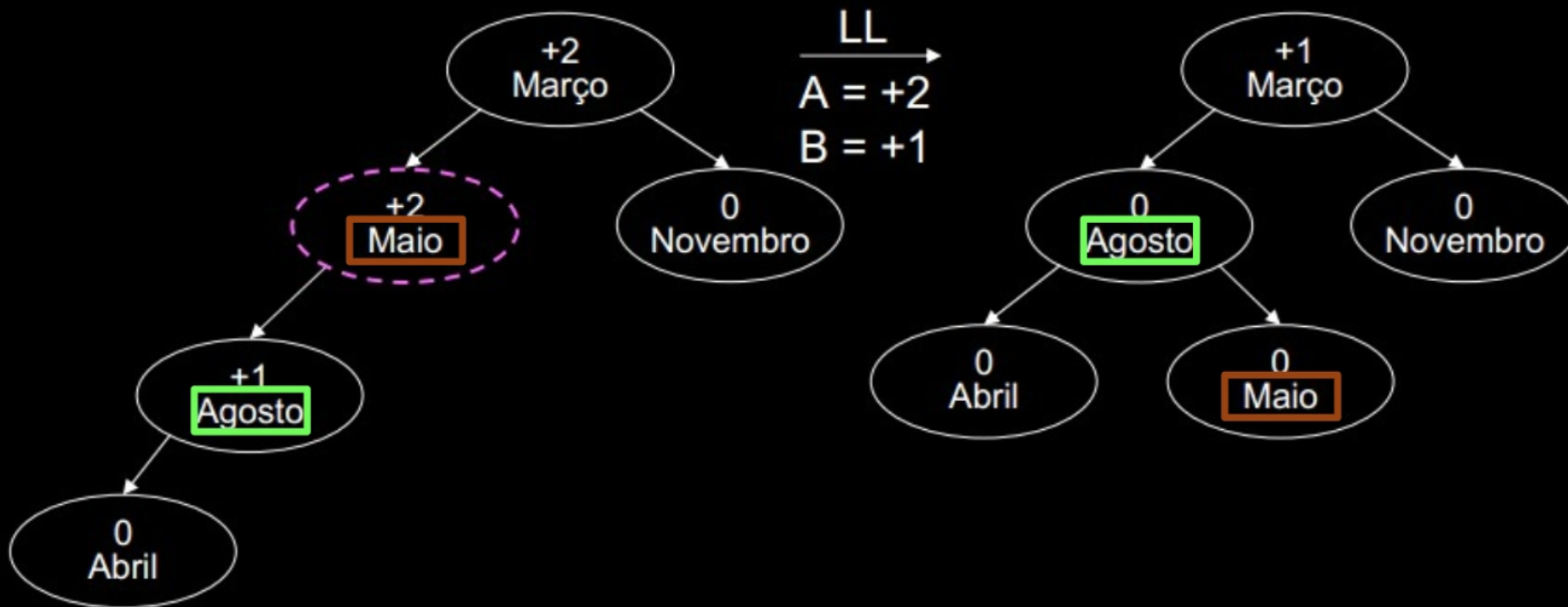
Depois do rebalanceamento

*Sem necessidade
de rebalanceamento*

Inserção de elementos - Abril

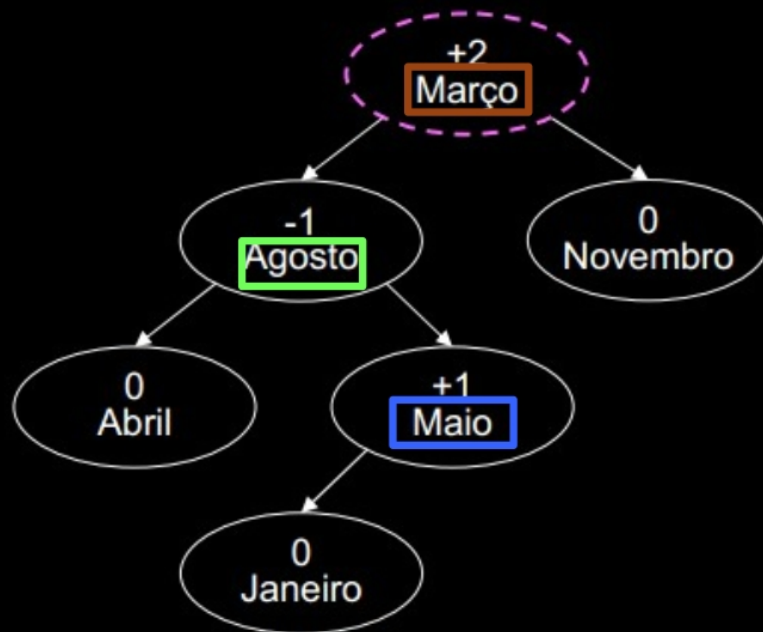
Depois da inserção

Depois do rebalanceamento



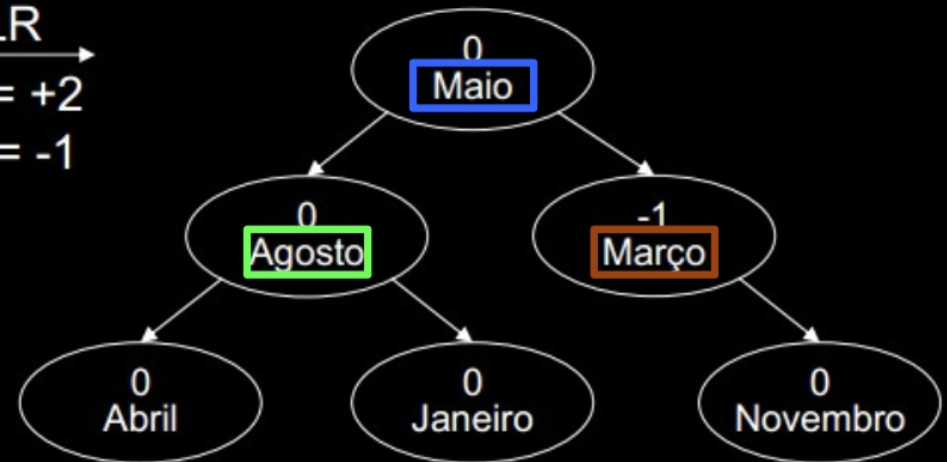
Inserção de elementos - Janeiro

Depois da inserção



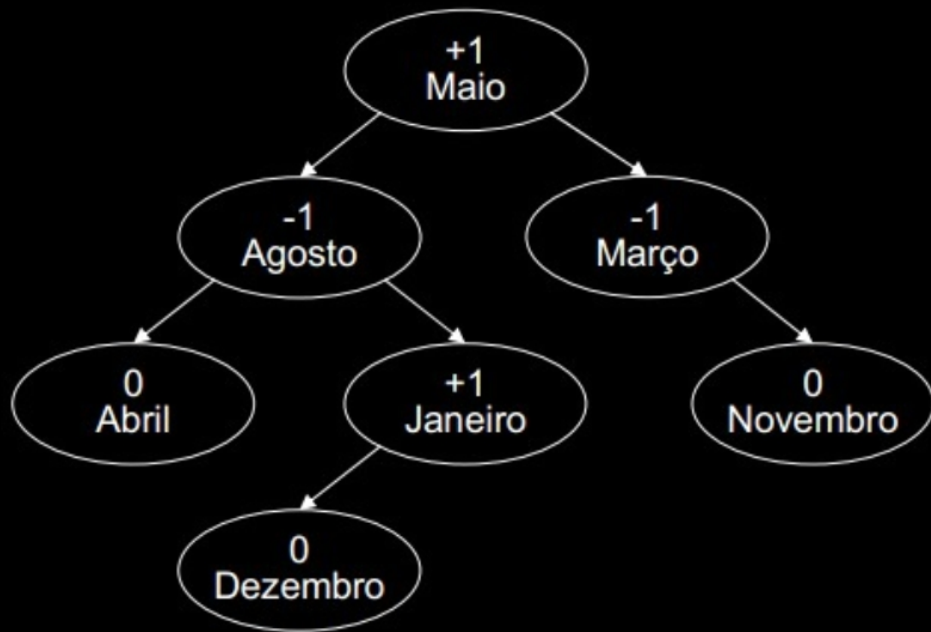
Depois do rebalanceamento

LR
A = +2
B = -1



Inserção de elementos - Dezembro

Depois da inserção



Depois do rebalanceamento

*Sem necessidade
de rebalanceamento*

Inserção de elementos - Julho

Depois da inserção

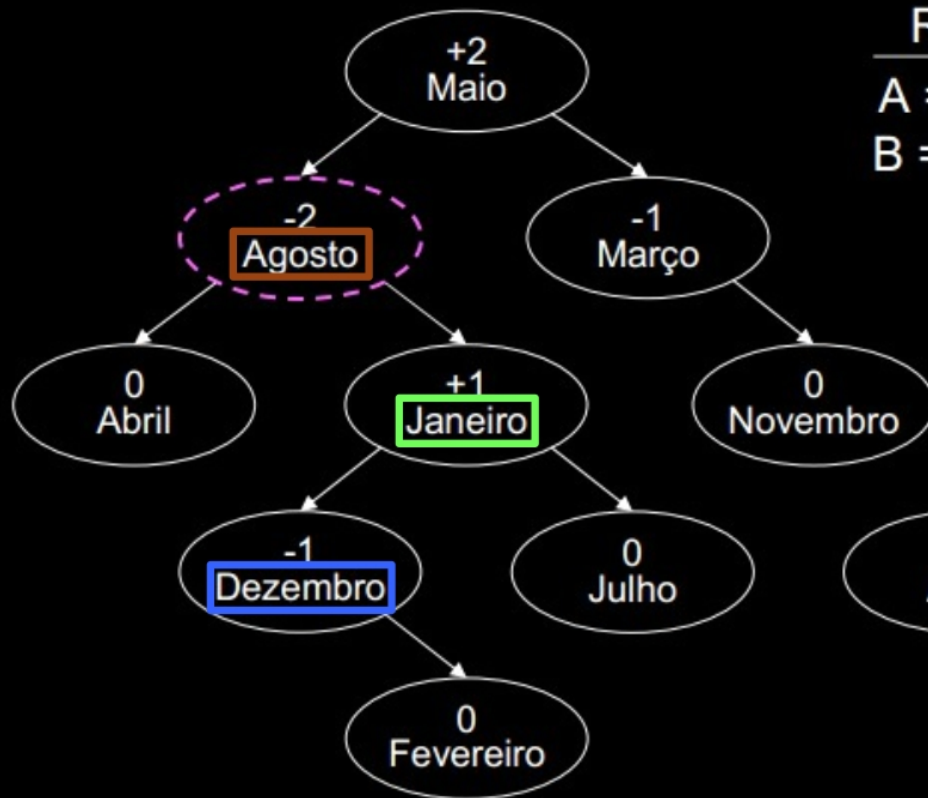


Depois do rebalanceamento

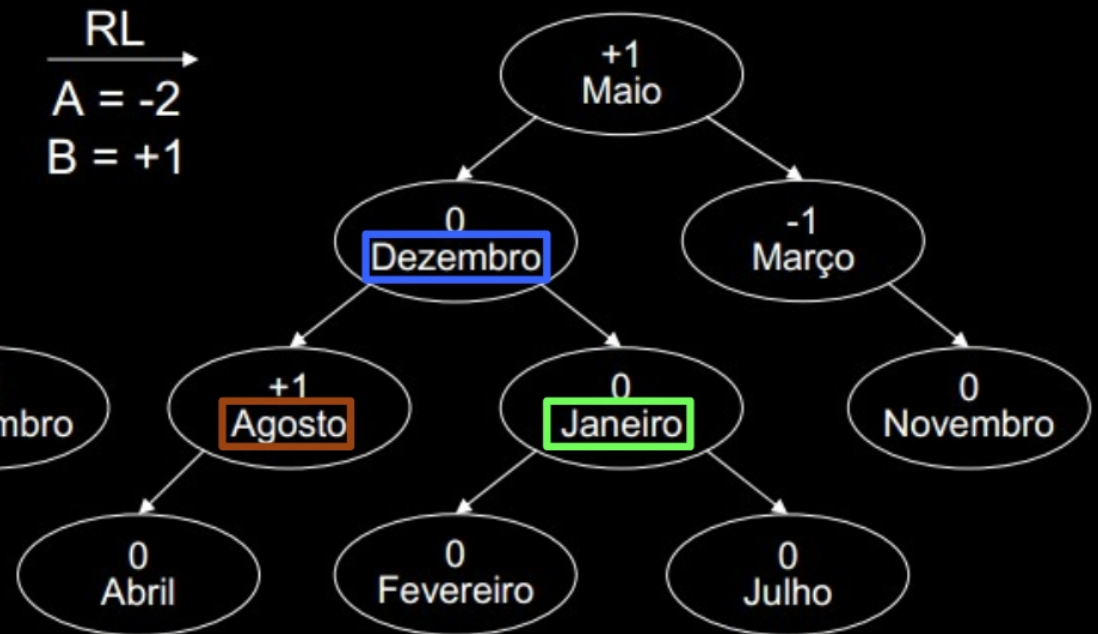
*Sem necessidade
de rebalanceamento*

Inserção de elementos - Fevereiro

Depois da inserção



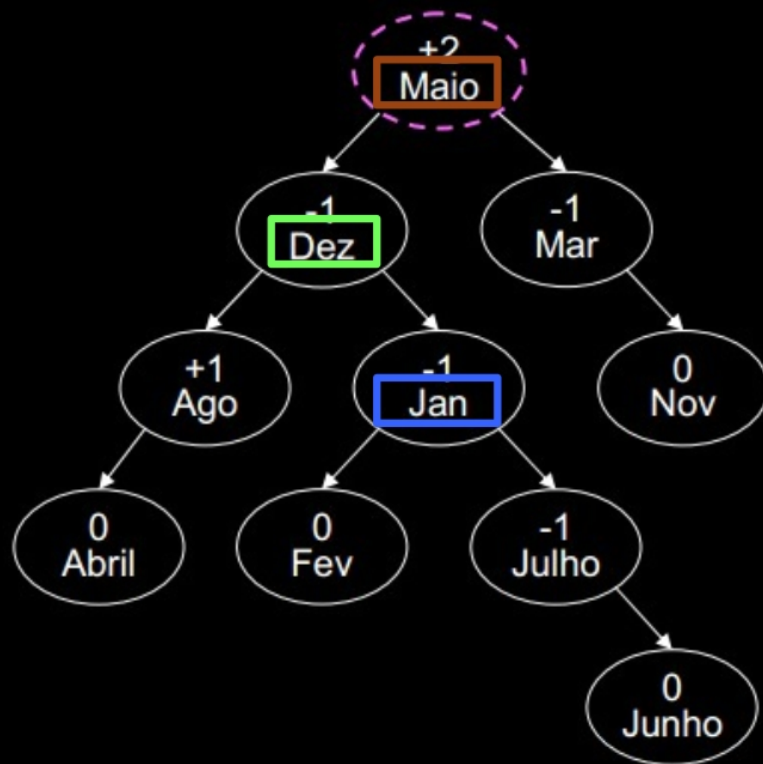
Depois do rebalanceamento



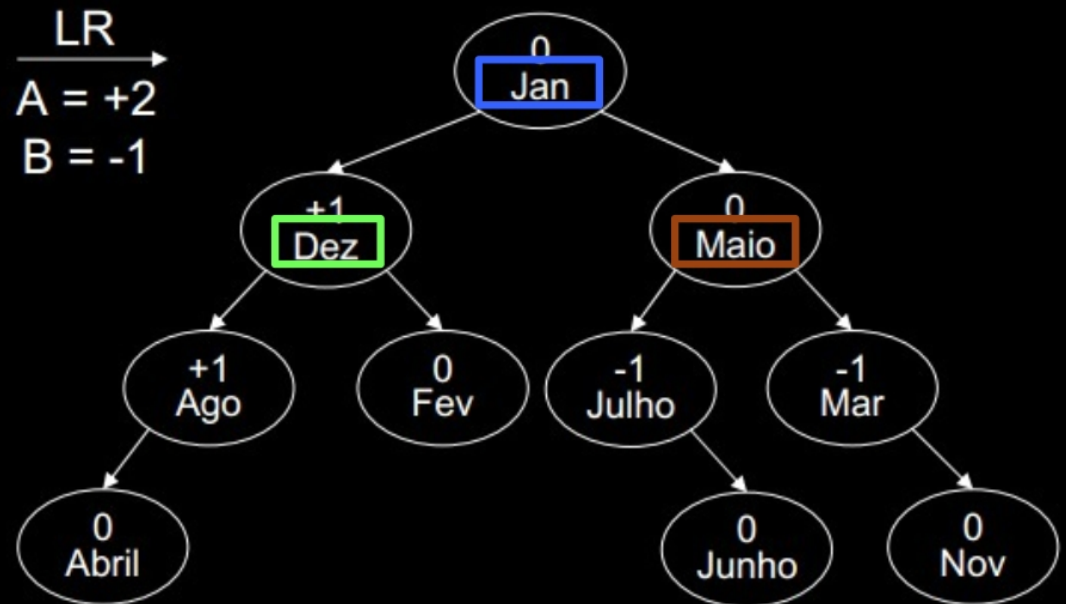
RL
A = -2
B = +1

Inserção de elementos - Junho

Depois da inserção

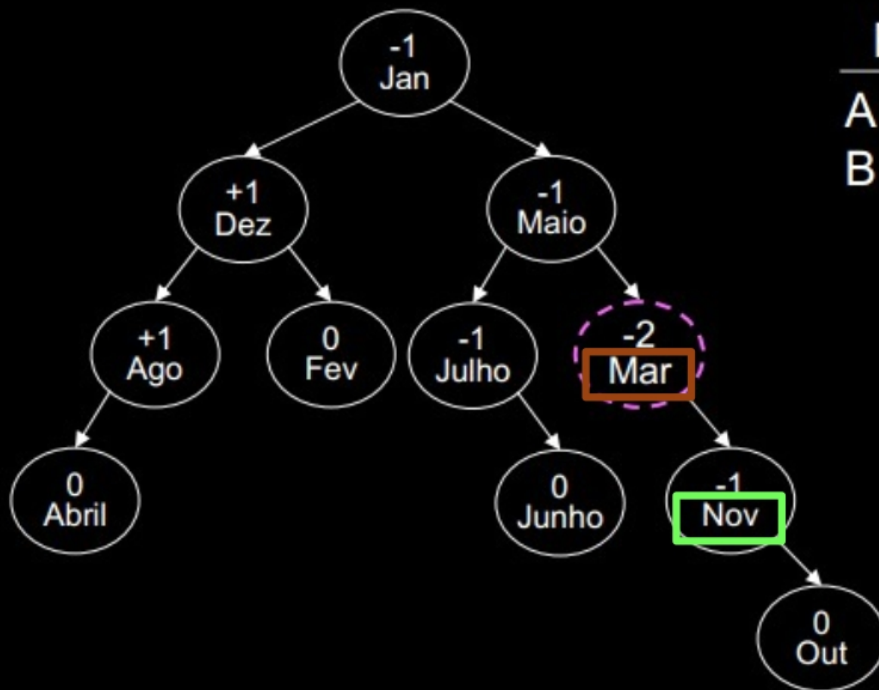


Depois do rebalanceamento



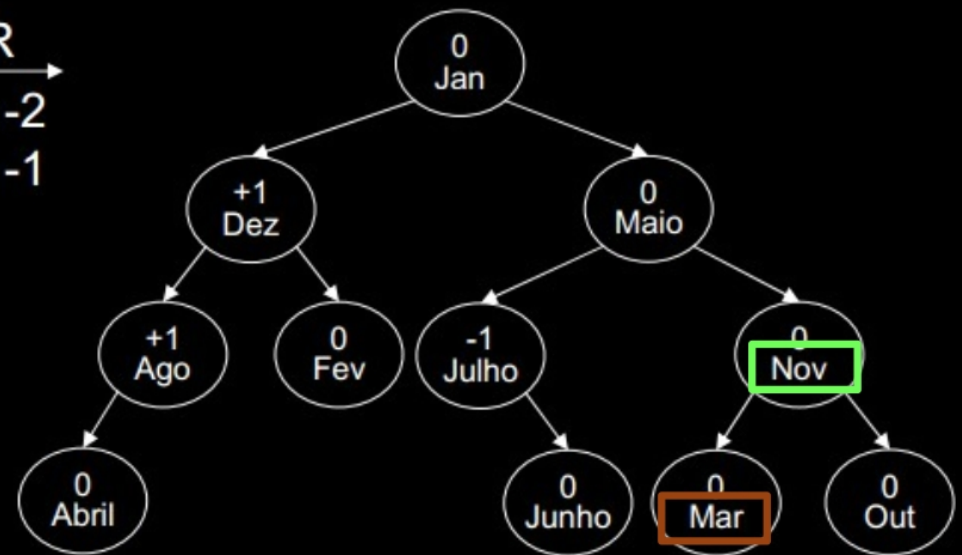
Inserção de elementos - Outubro

Depois da inserção



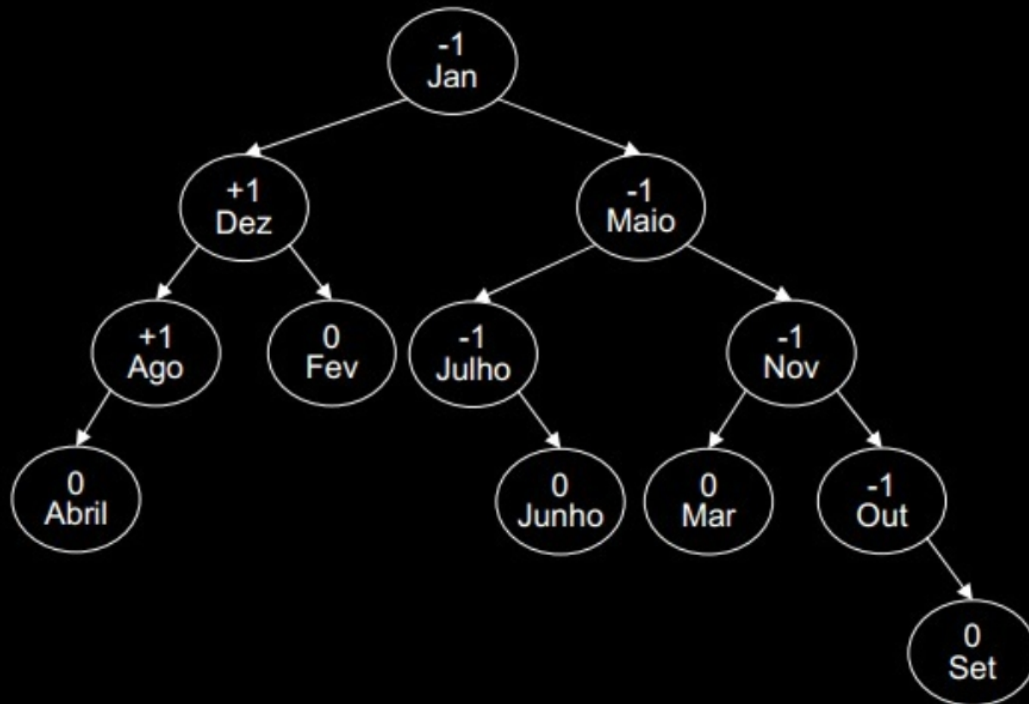
RR
A = -2
B = -1

Depois do rebalanceamento



Inserção de elementos - Setembro

Depois da inserção



Depois do rebalanceamento

*Sem necessidade
de rebalanceamento*

Árvore AVL - Rotações

O processo de rebalanceamento é conduzido utilizando 4 tipos de rotações

- LL
- RR
- LR
- RL

Suponha que o novo nó inserido é Y:

- As rotações são caracterizadas pelo ancestral A (com fator de balanceamento +2 ou -2) mais próximo do nó Y.
-
-

Árvore AVL - Rotações

Seja B o filho de A no qual ocorreu a inserção de Y

LL: Y inserido na subárvore esquerda da subárvore esquerda de A

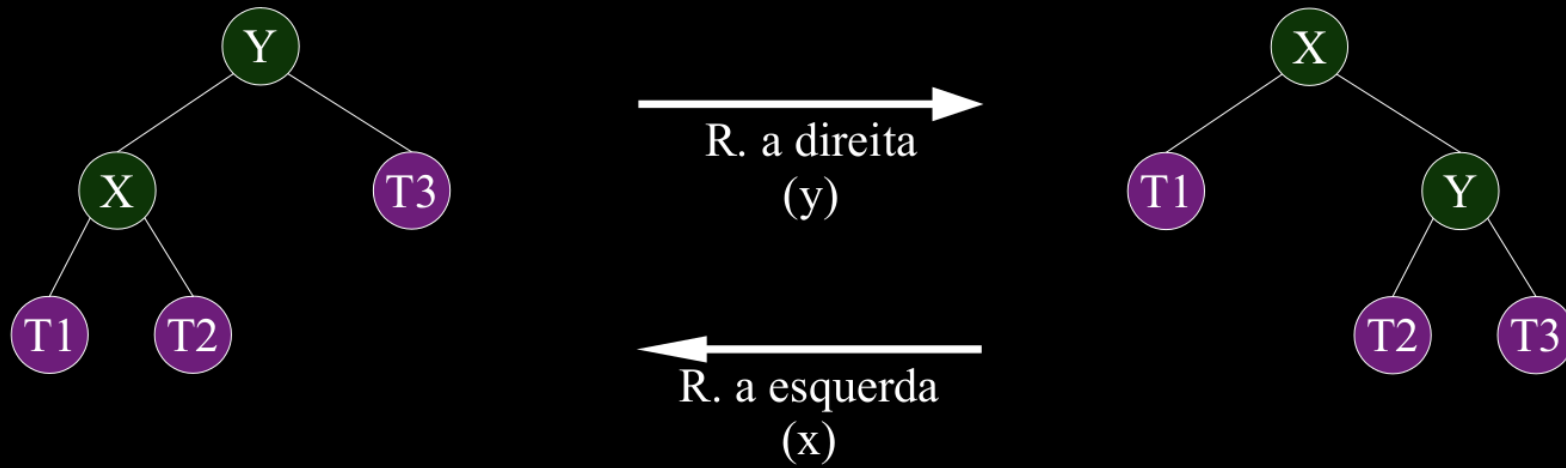
LR: Y inserido na subárvore direita da subárvore esquerda de A

RR: Y inserido na subárvore direita da subárvore direita de A

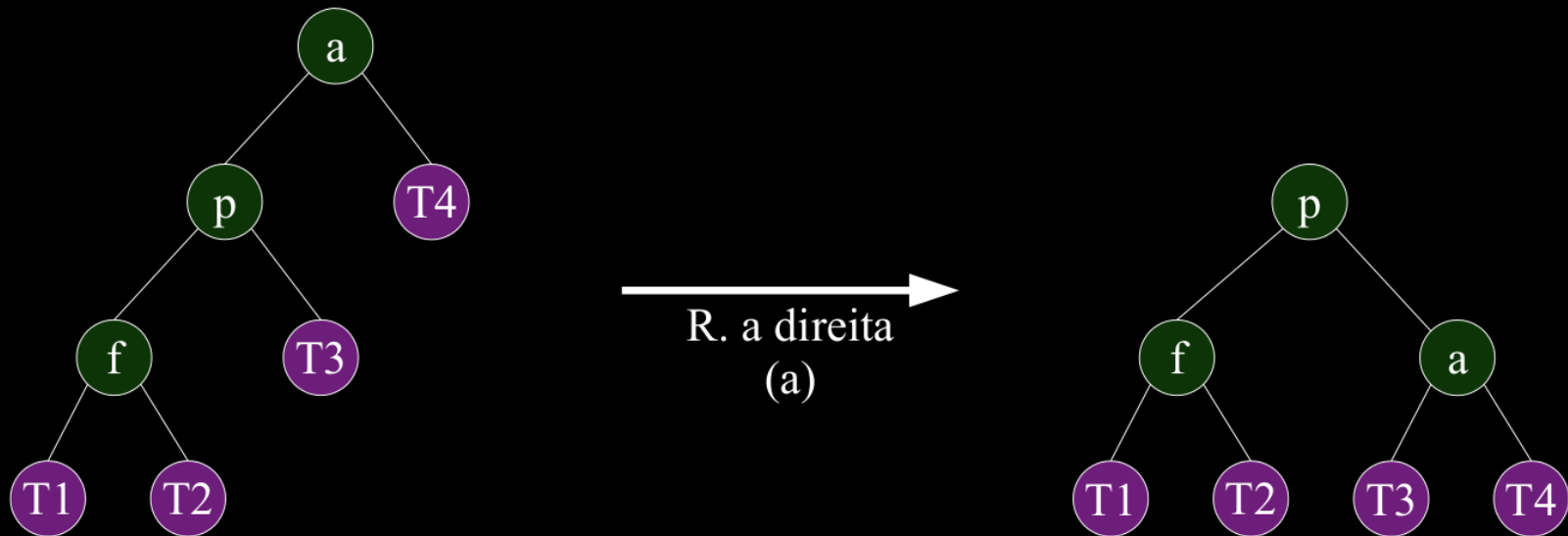
RL: Y inserido na subárvore esquerda da subárvore direita de A

- LL ($A = +2$; $B = +1$)
 - LR ($A = +2$; $B = -1$)
 - RR ($A = -2$; $B = -1$)
 - RL ($A = -2$; $B = +1$)
-
-

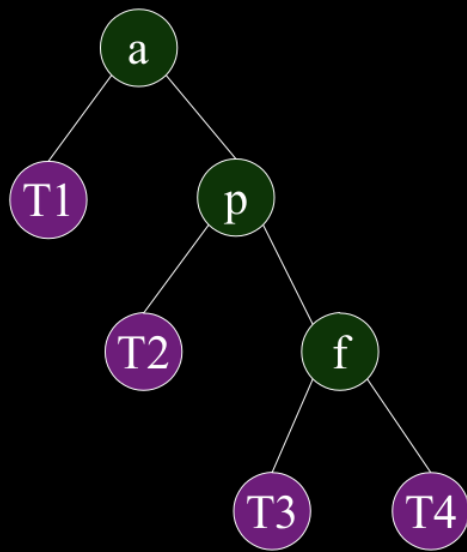
Árvore AVL - Rotações



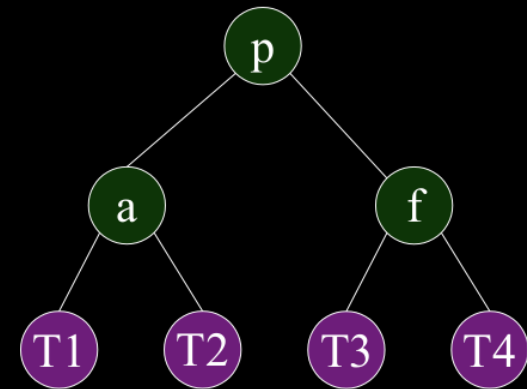
Árvore AVL – Rotações Caso LL



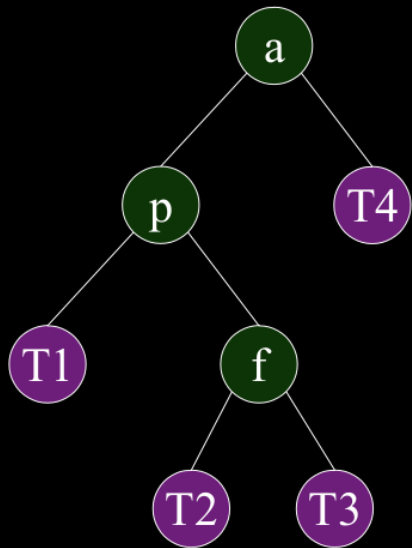
Árvore AVL – Rotações Caso RR



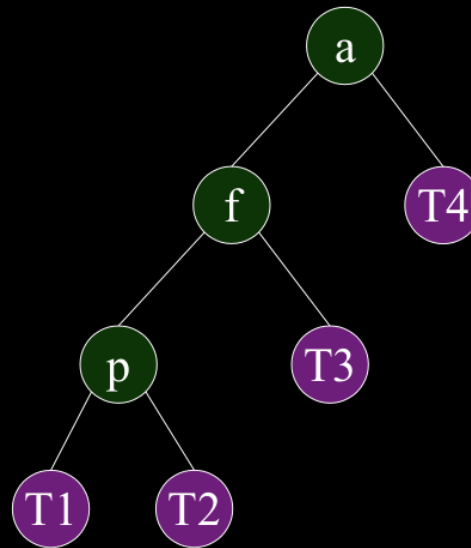
→
R. a esquerda
(a)



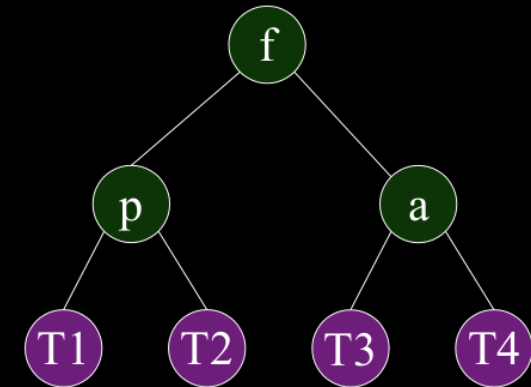
Árvore AVL – Rotações Caso LR



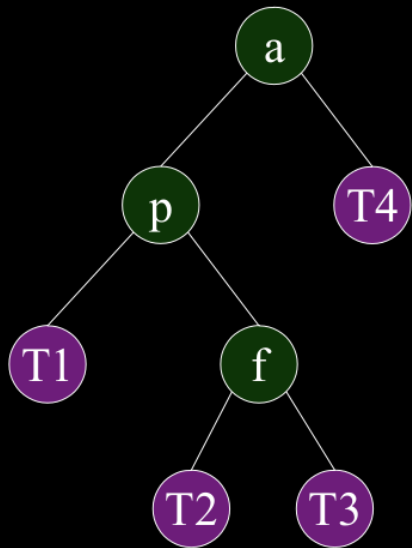
→
R. a esquerda
(p)



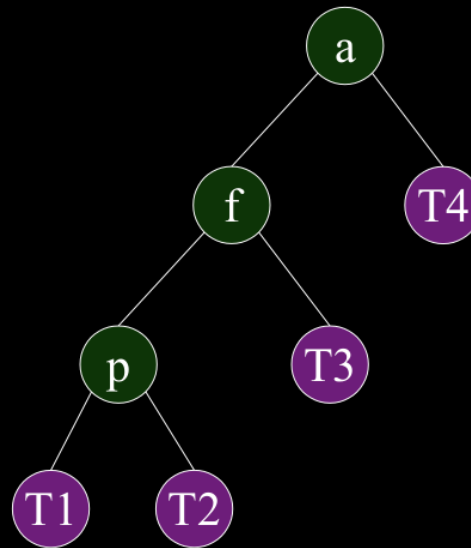
→
R. a direita
(a)



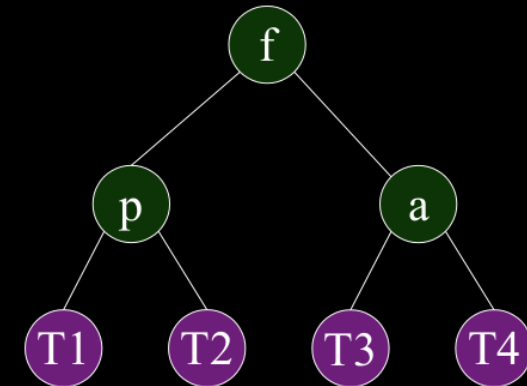
Árvore AVL – Rotações Caso RL



→
R. a esquerda
(p)



→
R. a direita
(a)

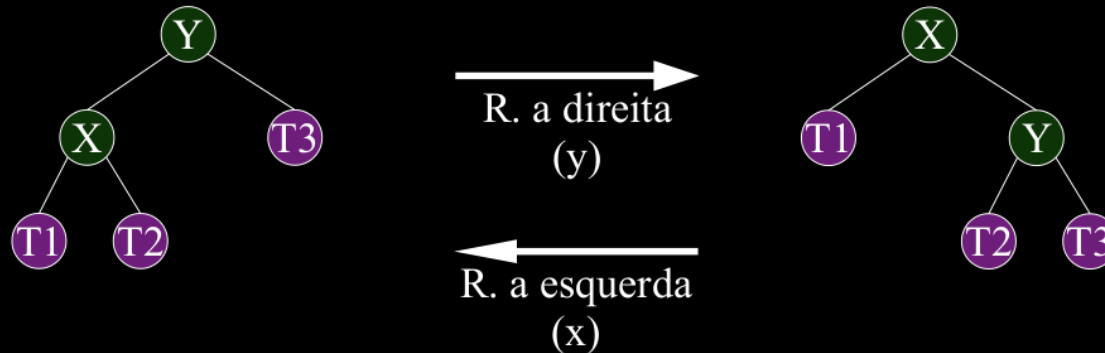


Árvore AVL – Modelagem

```
struct Node {  
    int data;  
    struct Node *left;  
    struct Node *right;  
    int height;  
};  
typedef struct Node Node;
```

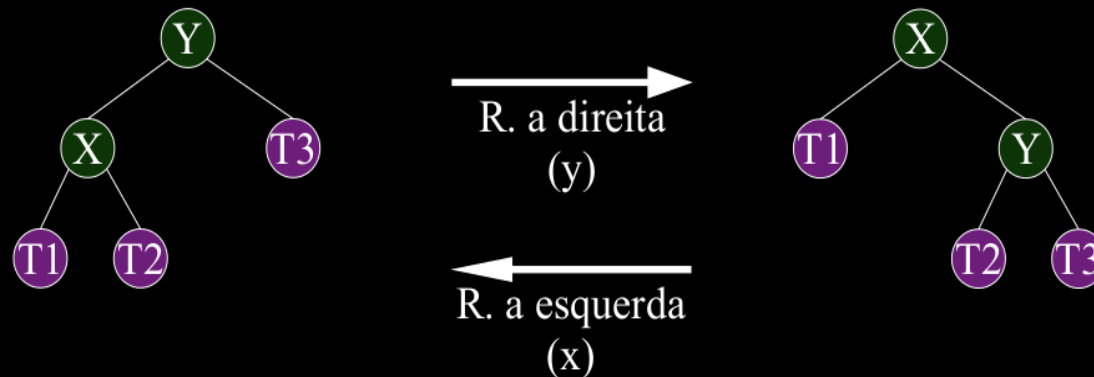
```
Node* inserirElemento(int valor, Node *raiz);  
Node* rotacaoADireita(Node* y);  
Node* rotacaoAESquerda(Node* x);  
void imprimirArvore(Node *raiz);  
void liberarArvore(Node* raiz);  
int fatorDeBalanceamento(Node* raiz);  
int altura(Node* raiz);  
int max(int a, int b);
```

Árvore AVL – Modelagem



```
Node* rotacaoADireita(Node* y) {  
    Node* x = y->left;  
    Node* T2 = x->right;  
  
    // rotacao  
    x->right = y;  
    y->left = T2;  
  
    // atualizacao de altura  
    y->height = 1 + max( altura(y->left) , altura(y->right) );  
    x->height = 1 + max( altura(x->left) , altura(x->right) );  
  
    return x;  
}
```

Árvore AVL – Modelagem



```
Node* rotacaoAEsquerda(Node* x) {  
    Node* y = x->right;  
    Node* T2 = y->left;  
  
    // rotacao  
    y->left = x;  
    x->right = T2;  
  
    // atualizacao de altura  
    x->height = 1 + max( altura(x->left) , altura(x->right) );  
    y->height = 1 + max( altura(y->left) , altura(y->right) );  
  
    return y;  
}
```

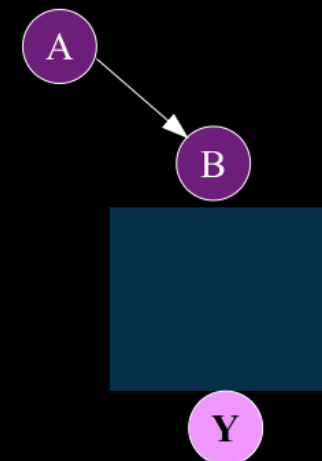
Árvore AVL – Inserção

```
Node* inserirElemento(int valor, Node *raiz) {  
    // insercao na ABB -- versao recursiva  
    if (raiz==NULL) {  
        Node *novo = (Node *) malloc(sizeof(Node));  
        novo->data = valor;  
        novo->left = NULL;  
        novo->right = NULL;  
        novo->height = 0;  
        return novo;  
    }  
  
    if (valor == raiz->data)  
        return raiz;  
    else {  
        if (valor < raiz->data)  
            raiz->left = inserirElemento(valor, raiz->left);  
        else  
            raiz->right = inserirElemento(valor, raiz->right);  
    }  
  
    // atualizacao da altura do Node antecessor  
    raiz->height = 1 + max( altura(raiz->left), altura(raiz->right) );  
  
    // fator de balanceamento  
    int fb = fatorDeBalanceamento(raiz);
```

Árvore AVL – Inserção

```
// -----  
// se o Node estiver desbalanceado, entao temos 4 alternativas  
  
// Caso LL:  
if ( fb>1 && valor < raiz->left->data) {  
    printf("\n* Rotacao LL");  
    return rotacaoADireita(raiz);  
}  
  
// Caso RR:  
if ( fb<-1 && valor > raiz->right->data) {  
    printf("\n* Rotacao RR");  
    return rotacaoAEsquerda(raiz);  
}  
  
// Caso LR:  
if ( fb>1 && valor > raiz->left->data) {  
    printf("\n* Rotacao LR");  
    raiz->left = rotacaoAEsquerda(raiz->left);  
    return rotacaoADireita(raiz);  
}  
  
// Caso RL:  
if ( fb<-1 && valor < raiz->right->data) {  
    printf("\n* Rotacao RL");  
    raiz->right = rotacaoADireita(raiz->right);  
    return rotacaoAEsquerda(raiz);  
}  
  
return raiz;  
}
```

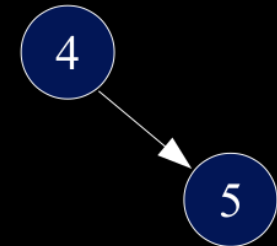
LL (A = +2; B = +1)
LR (A = +2; B = -1)
RR (A = -2; B = -1)
RL (A = -2; B = +1)



Árvore AVL – Atividade

Suponha que serão realizadas as seguintes inserções de chaves na árvore AVL ao lado:

- 7
- 2
- 1
- 3
- 6



- (a) Apresente a árvore AVL resultante (1 árvore)
 - (b) Indique o tipo de rotações consideradas em cada inserção
-
-

AULA 16

ESTRUTURA DE DADOS

Árvores AVL

