Fundamentos de Algoritmos e Estrutura de Dados
#4 – Estruturas Não Lineares - Árvores

Prof. André Gustavo Hochuli

gustavo.hochuli@pucpr.br
aghochuli@ppgia.pucpr.br

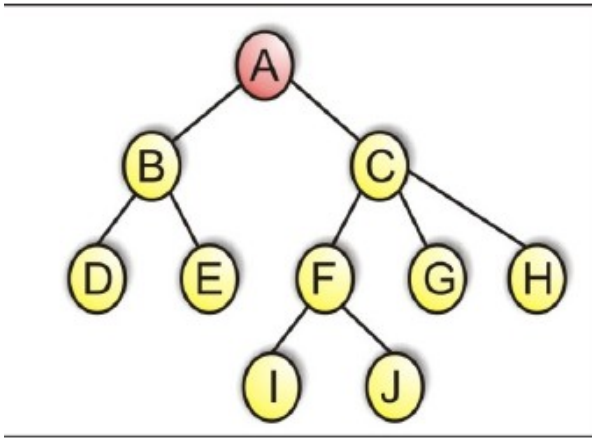
Plano de Aula

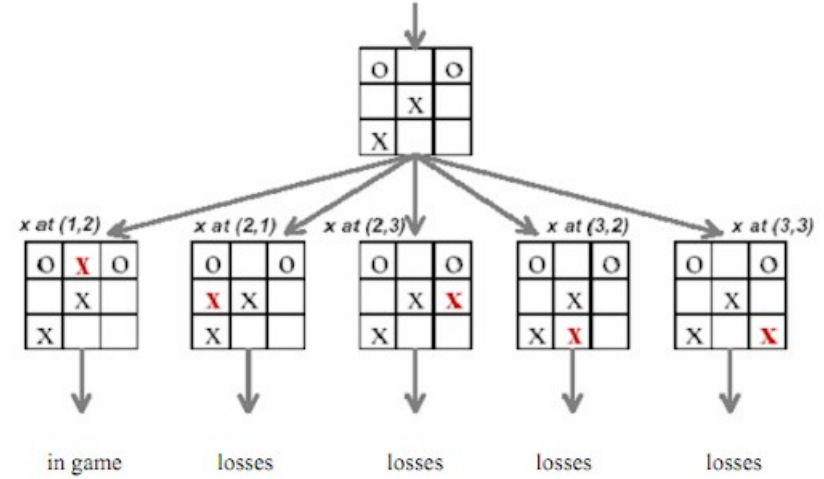
- Estrutura de Dados Não Lineares
  - Conceito de Árvores
  - Árvores Binárias
  - Árvores Binárias Balancedeadas (AVL - Adelson-Velsky and Landis)

Árvores

Árvores

- **Estrutura não linear**
- **Representação Hierárquica**
- **Aplicações**
  - **Verificadores de sintaxe (Grammar, Compiladores)**
  - **Banco de Dados**
  - **Roteadores**
  - **Escalonadores de processos**
  - **I.A**



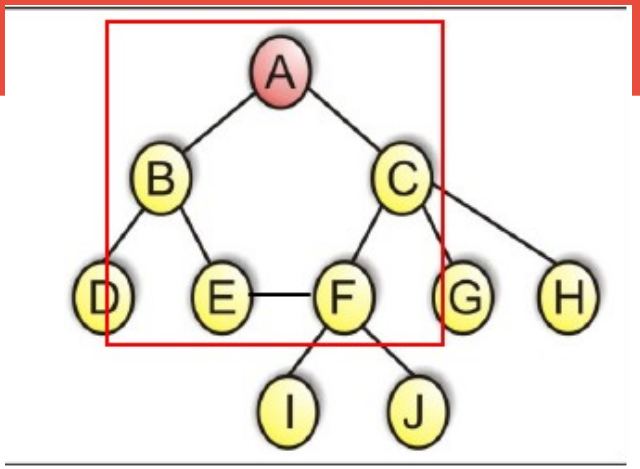


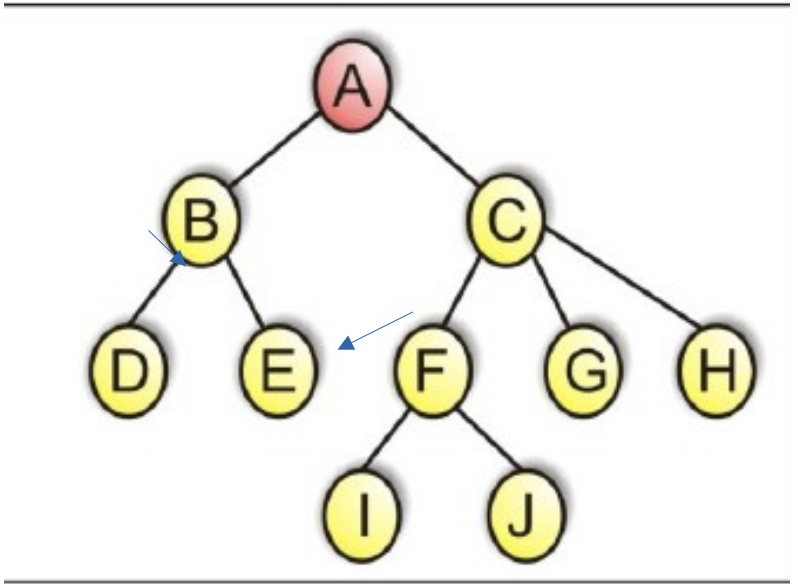
Árvores - Conceitos

- Árvore não contém ciclos

- Grau:
  - Número de sub-árvores
    - Nós: A=2, C=3, D=0
    - Grau Árvore: 3

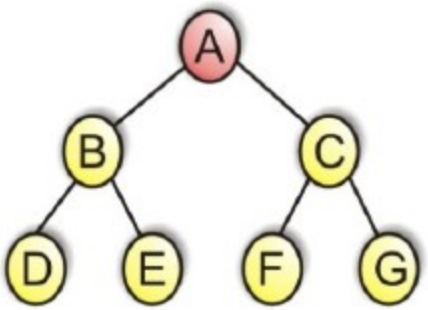
- Nível
  - Distância entre o vértice até a raiz
    - Nós: D=2, I=3
    - Nível da Árvore: 3

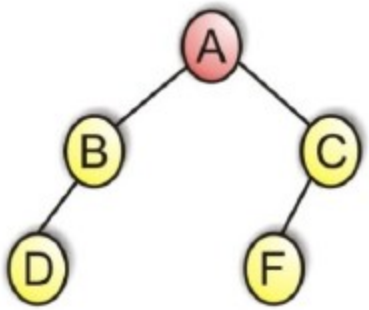




Árvores - Binárias

- **Árvores Binárias tem grau 2**

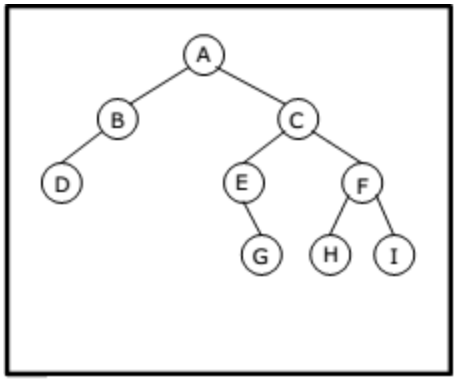




- **Caminhamento**
  - **Pré-Ordem: raiz→esq→dir**
  - **Pós-Ordem: esq →dir→raiz**
  - **In-Ordem: esq→raiz→dir**
  - **Nível*: raizes(N=0)→raizes(N=1)....**

    (*) Método não recursivo

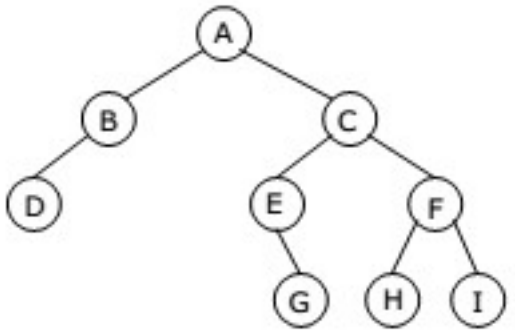
    (*) Árvore deve ser convertido em fila



- Preorder
  A, B, D, C, E, G, F, H, I
- Postorder
  D, B, G, E, H, I, F, C, A
- Inorder
  D, B, A, E, G, C, H, F, I
- Level order
  A, B, C, D, E, F, G, H, I

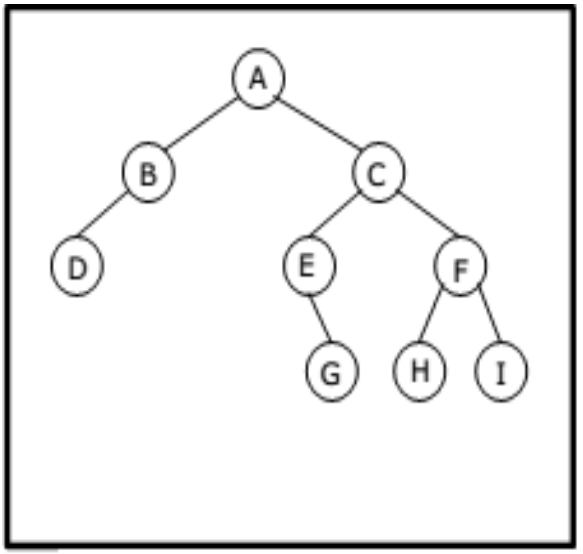
Árvores - Implementação

```
class Node:
    def __init__(self,data):
        self.data = data  # Assign data
        self.left = None  # Initialize as None
        self.right = None  # Initialize as None

class Binary_Tree:

    # Init Class
    def __init__(self,data):
        self.root = Node(data)

    def push(self,data):

        if self.root is None:
            print("Root")
            self.root = Binary_Tree(data)

        if data > self.root.data:
            if self.root.right is None:
                print("Add Right")
                self.root.right = Binary_Tree(data)
            else:
                self.root.right.push(data)

        else:
            if self.root.left is None:
                print("Add Left")
                self.root.left = Binary_Tree(data)
            else:
                self.root.left.push(data)

        return
```



Árvores - Caminhamento

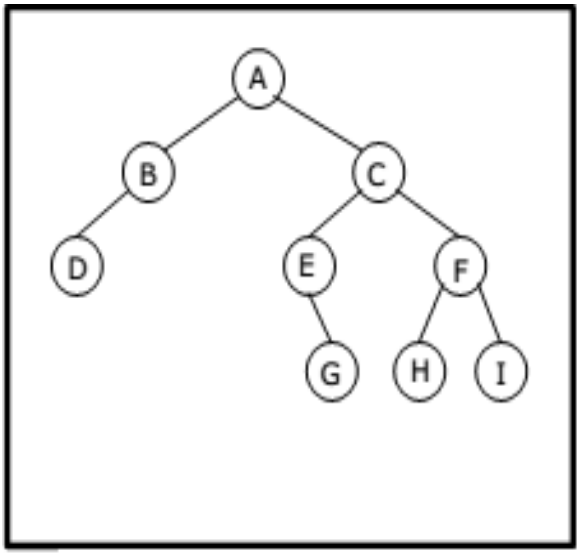
```
def walk_preorder(self):

    print(self.root.data)

    if self.root.left is not None:
        self.root.left.walk_preorder()

    if self.root.right is not None:
        self.root.right.walk_preorder()
```



- Preorder
  A, B, D, C, E, G, F, H, I
- Postorder
  D, B, G, E, H, I, F, C, A
- Inorder
  D, B, A, E, G, C, H, F, I
- Level order
  A, B, C, D, E, F, G, H, I

Árvores - Exercício

- **Implemente:**
  - **Post-Order**
  - **In-Order**



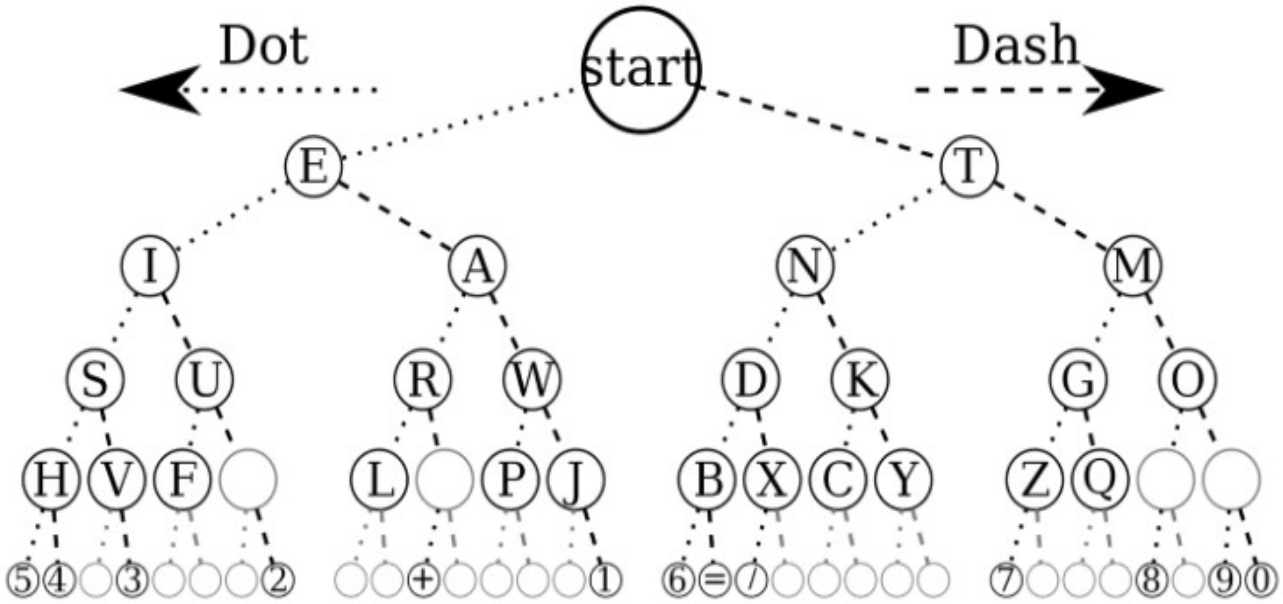
- Preorder
  A, B, D, C, E, G, F, H, I
- Postorder
  D, B, G, E, H, I, F, C, A
- Inorder
  D, B, A, E, G, C, H, F, I
- Level order
  A, B, C, D, E, F, G, H, I

Árvores - Exercício

Implementação de um tradutor de código morse

A	·−	J	·−−−	S	···	2	··−−−
B	−···	K	−·−	T	−	3	···−−
C	−·−·	L	·−··	U	··−	4	····−
D	−··	M	−−	V	···−	5	·····
E	·	N	−·	W	·−−	6	−····
F	··−·	O	−−−	X	−··−	7	−−···
G	−−·	P	·−−·	Y	−·−−	8	−−−··
H	····	Q	−−·−	Z	−−··	9	−−−−·
I	··	R	·−·	1	·−−−−	0	−−−−−

--- .-.. .-/-- ..- -. -.. --- = Ola Mundo



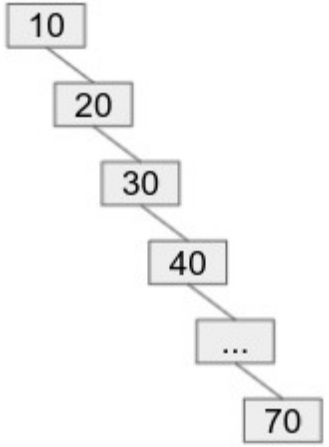
Árvores AVL

Árvores Binárias Balanceadas (AVL)

- **Adelson Velsy e Landis (AVL)**

Adelson-Velsky, Georgy; Landis, Evgenii (1962). "An algorithm for the organization of information". Proceedings of the USSR Academy of Sciences (in Russian). 146: 263–266

- **Inclusão sequêncial em Binary Search Tree (BST)**
  - **10,20,30,40,50,60,70**



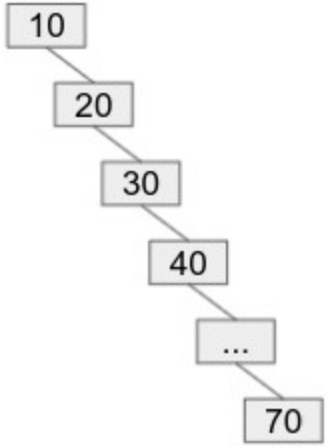
Desbalanceada

Árvores Binárias Balanceadas (AVL)

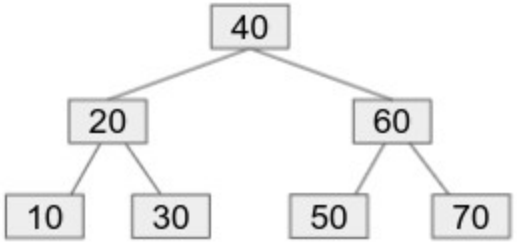
- **Adelson Velsy e Landis (AVL)**

Adelson-Velsky, Georgy; Landis, Evgenii (1962). "An algorithm for the organization of information". Proceedings of the USSR Academy of Sciences (in Russian). 146: 263–266

- **Inclusão sequêncial em Binary Search Tree (BST)**
  - **10,20,30,40,50,60,70**



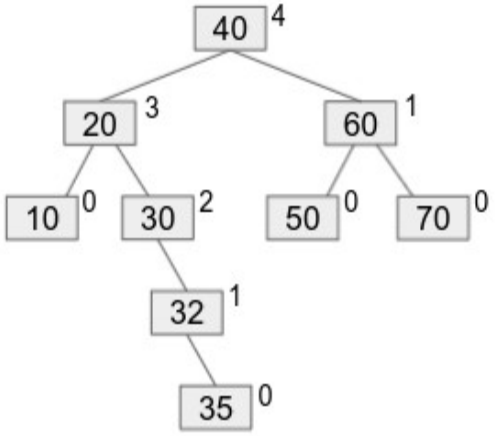
Desbalanceada



Balanceada

Árvores Binárias Balanceadas (AVL)

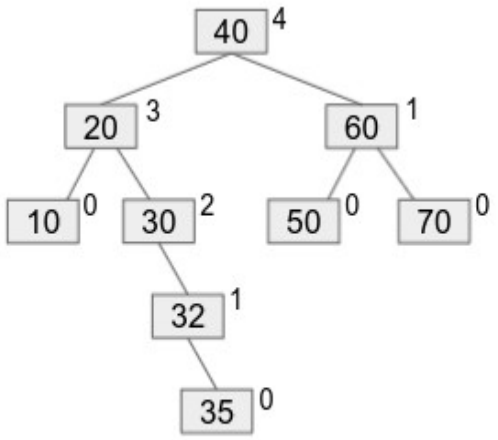
- **Árvore balanceada: altura do lado esquerdo da árvore não difere mais de +-1 do lado direito.**
- **Ou seja, Fator de Balanceamento é dado por:**
  - **FB( ) = he - hd → |FB| <= 1 == Nodo balanceado**
    - **FB( ) == 0 → he == hd**
    - **FB( ) > 0 → he > hd**
    - **FB( ) < 0 → he < hd**



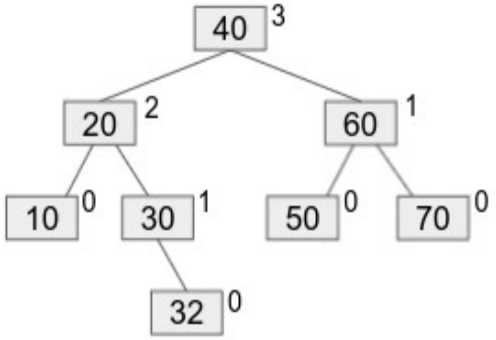
Desbalanceada

Árvores Binárias Balanceadas (AVL)

- **Árvore balanceada: altura da árvore esquerda não difere mais de +-1 da árvore esquerda.**
- **Ou seja, Fator de Balanceamento é dado por:**
  - **FB( ) = he - hd → |FB| <= 1 == Nodo balanceado**
    - **FB( ) == 0 → he == hd**
    - **FB( ) > 0 → he > hd**
    - **FB( ) < = → he < hd**

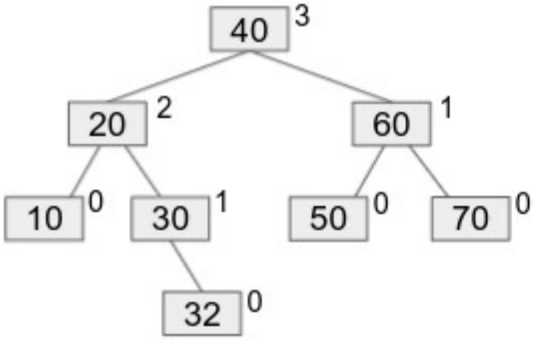


Desbalanceada



Balanceada

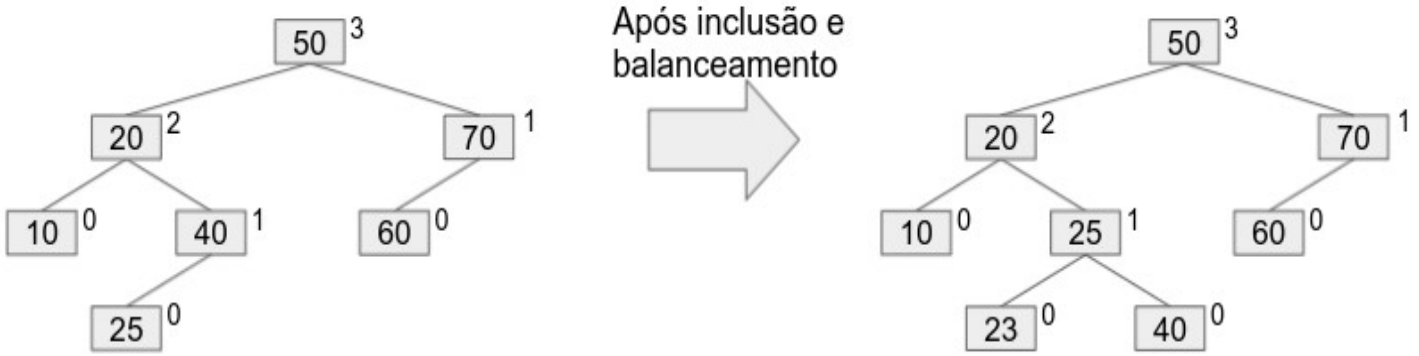
Árvores Binárias Balanceadas (AVL)




```
class Node:
    def __init__(self,data):
        self.data = data  # Assign data
        self.left = None  # Initialize as None
        self.right = None  # Initialize as None
        self.height  = 1
```

Árvores Binárias Balanceadas (AVL)

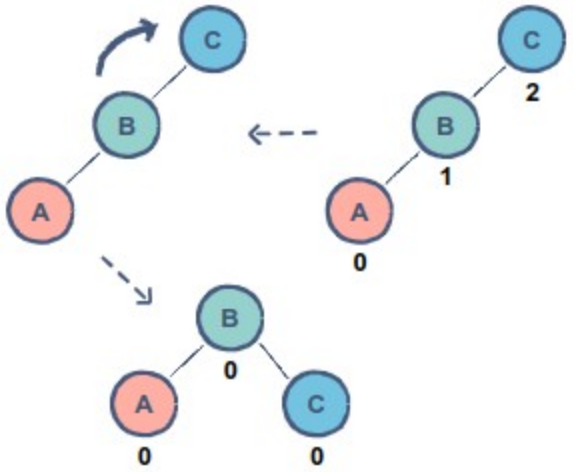
- **Inclusões:**
  - **Inserir 23**
  - **Balancear se |FB| > 1**



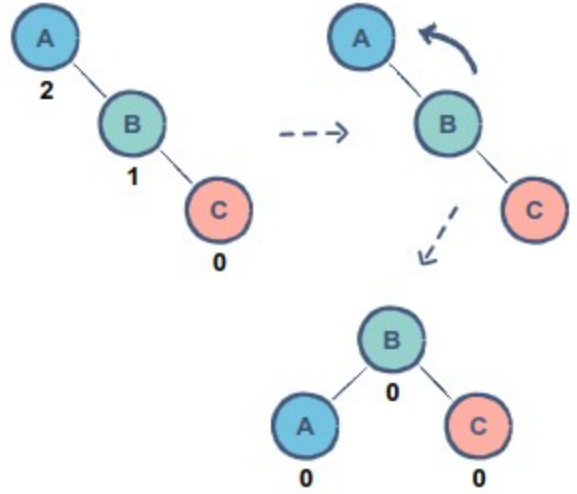
  - **Como definir rotações?**
    - **A esquerda ou Direita?**
    - **Dupla ?**

Árvores Binárias Balanceadas (AVL)

- **Rotação a Direita**
  - **FB > 1 e valor inserido a esquerda**

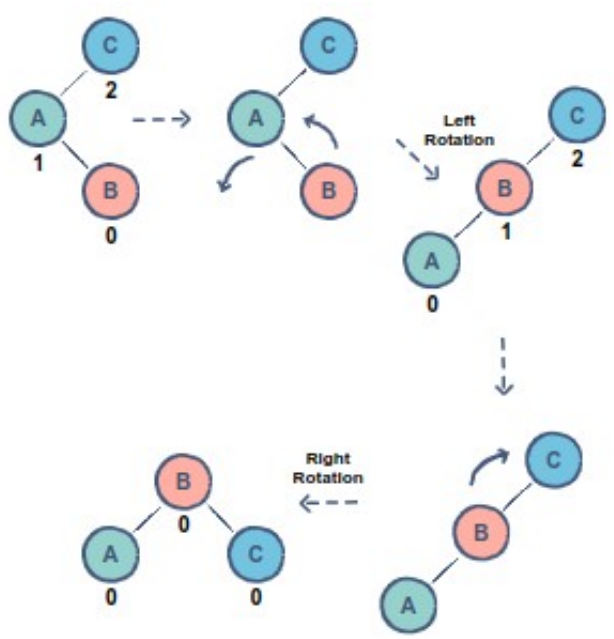


- **Rotação a Esquerda**
  - **FB < -1 e valor inserido a direita**

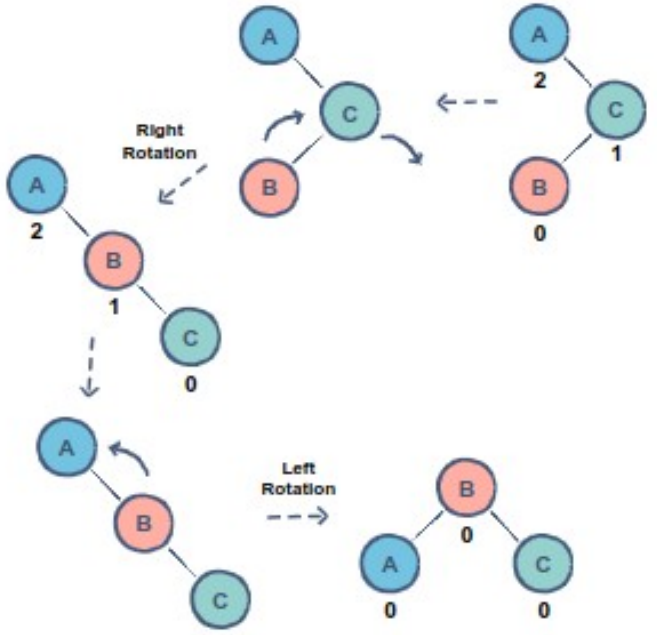


Árvores Binárias Balanceadas (AVL)

- **Rotação Dupla Esq-Dir**
  - **FB > 1 e valor inserido a direita**

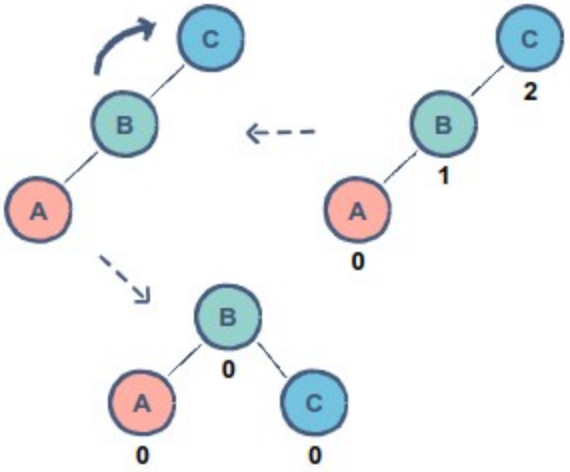


- **Rotação Dupla Dir-Esq**
  - **FB < -1 e valor inserido a esquerda**



Árvores Binárias Balanceadas (AVL)

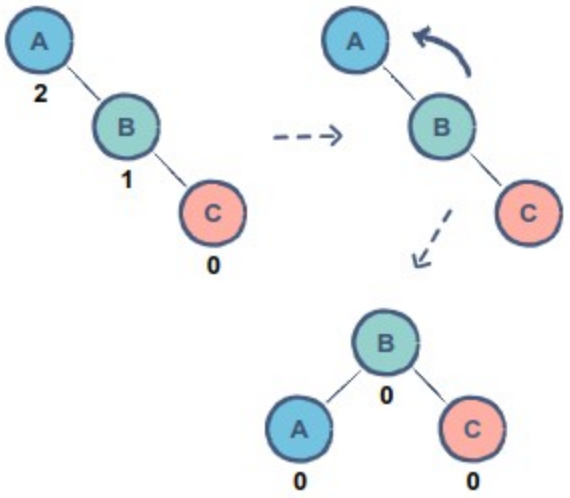
- **Pseudocódigos**
- **Rotação a Direita**
  - **FB > 1 e valor inserido a esquerda**




```
if (balance > 1 && key < node->left->key)
    node = rightRotate(node);


Node *rightRotate(Node *y)
{
    Node *x = y->left;
    Node *T2 = x->right;

    // Perform rotation
    x->right = y;
    y->left = T2;

    // Update heights
    y->height = max(height(y->left),
                    height(y->right)) + 1;
    x->height = max(height(x->left),
                    height(x->right)) + 1;

    // Return new root
    return x;
}
```

Árvores Binárias Balanceadas (AVL)

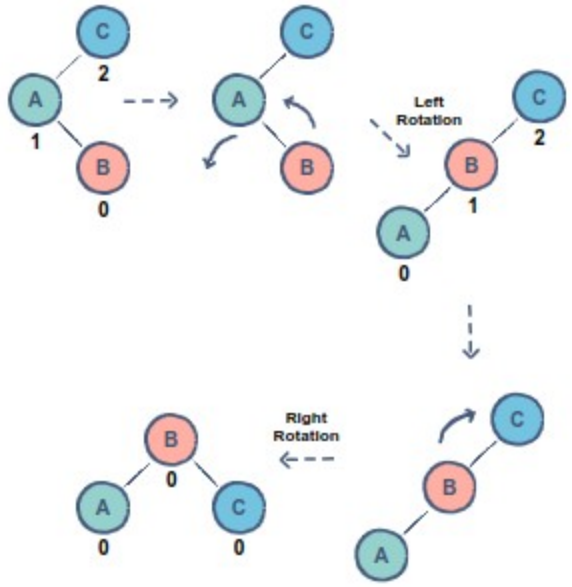
- **Rotação a Esquerda**
  - **FB < -1 e valor inserido a direita**




```
if (balance < -1 && key > node->right->key)
    node =  leftRotate(node);


Node *leftRotate(Node *x)
{
    Node *y = x->right;
    Node *T2 = y->left;

    // Perform rotation
    y->left = x;
    x->right = T2;

    // Update heights
    x->height = max(height(x->left),
                    height(x->right)) + 1;
    y->height = max(height(y->left),
                    height(y->right)) + 1;

    // Return new root
    return y;
}
```

Árvores Binárias Balanceadas (AVL)

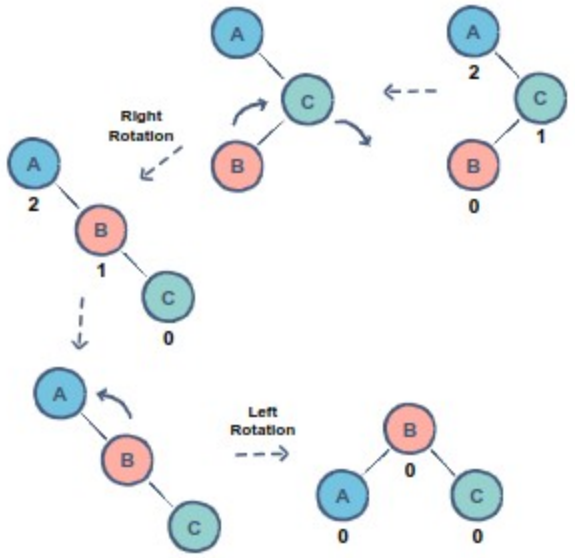
- **Rotação Dupla Esq-Dir**
- **FB > 1 e valor inserido a direita**




```
if (balance > 1 && key > node->left->key)
{
    node->left = leftRotate(node->left);
    node = rightRotate(node);
}
```

Árvores Binárias Balanceadas (AVL)

- **Rotação Dupla Dir-Esq**
  - **FB < -1 e valor inserido a esquerda**




```
if (balance < -1 && key < node->right->key)
{
    node->right = rightRotate(node->right);
    return leftRotate(node);
}
```

Árvores Binárias Balanceadas (AVL)

- **Exercício (Manualmente)**
  - **Inserir as sequências abaixo, utilizando arvores AVL e não-AVL**
    - **10, 3, 2, 5, 7 e 6**
    - **A,B,C ..... J**
  - **Caminhos em in-ordem em ambas as arvores**
  - **Número de buscas necessárias para encontrar o elemento 7 e H**
    - **AVL e Não-Balanceada**
  - **BigO ? Porque?**
  - **Simulador AVL:**
    - https://cmps-people.ok.ubc.ca/ylucet/DS/AVLtree.html
    - https://visualgo.net/en

Trabalho

Trabalho 01: Análise Comparativa de Estruturas de Dados

Trabalho 01: Análise Comparativa de Estruturas de Dados

O trabalho é formalizado no documento PDF abaixo anexado:

https://drive.google.com/file/d/1iNAsY1hraWFMaJjWwBbhne1AnEuk6lHr/view?usp=drive_link

Peso: 30%

Pontos 10
Enviando um upload de arquivo
Tipos de arquivo zip

Entrega	Para	Disponível a partir de	Até
14 set	Todo mundo	-	14 set em 23:59