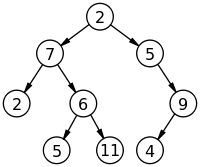
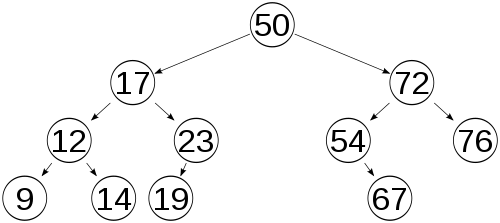
1-

**Primeiramente, destacasse os conceitos de arvore binaria e AVL.**  
  
**Árvore binária ->** É uma estrutura de dados caracterizada por:  
Ou não tem elemento algum (árvore vazia).  
Ou tem um elemento distinto, denominado raiz, com dois apontamentos para duas estruturas diferentes, denominadas sub-árvore esquerda e sub-árvore direita.  
Perceba que a definição é recursiva e, devido a isso, muitas operações sobre árvores binárias utilizam recursão. É o tipo de árvore mais utilizado na computação. A principal utilização de árvores binárias são as árvores de busca.  
Os nós de uma árvore binária possuem graus zero, um ou dois. Um nó de grau zero é denominado folha.  
Uma árvore binária é considerada estritamente binária se cada nó da árvore possui grau zero ou dois. A profundidade de um nó é a distância deste nó até a raiz. Um conjunto de nós com a mesma profundidade é denominado nível da árvore. Denominamos altura, a maior profundidade existente na árvore, ou seja, a profundidade do nó mais profundo. Uma árvore é dita completa se todas as folhas da árvore estão no mesmo nível da árvore.



**Árvore AVL ->** É uma [árvore binária de busca](https://pt.wikipedia.org/wiki/%C3%81rvore_bin%C3%A1ria_de_busca) balanceada, ou seja, uma árvore balanceada (árvore completa) são as árvores que minimizam o número de comparações efetuadas no pior caso para uma busca com chaves de probabilidades de ocorrências idênticas. Contudo, para garantir essa propriedade em aplicações dinâmicas, é preciso reconstruir a árvore para seu estado ideal a cada operação sobre seus nós (inclusão ou exclusão). Nessa estrutura de dados cada elemento é chamado de nó. Cada nó armazena uma chave e dois ponteiros, uma para a sub-árvore esquerda e outro para a sub-árvore direita.



**Segundamente, abordando as suas diferenças.**  
  
Uma árvore binária balanceada (AVL) é uma árvore binária na qual as alturas das duas sub-árvores de todo nó nunca diferem em mais de 1. Uma árvore binária de busca depende da ordem da inserção para ter um tempo assintótico de busca ótimo, visto que o primeiro valor inserido será usado como uma raiz e os demais irão para esquerda ou para direita se forem maiores ou menores. Sendo assim se você adicionar os valores em ordem crescente de s ficarão todos a direita do valor anterior, logo o tempo de busca será de O(n), sendo n o número de valores. Já em uma árvore AVL isso não ocorre, pois, cada valor na árvore possui um dado que determina seu balanceamento baseado na altura do seu nó a direita menos a altura do seu nó a esquerda, lembrando que esses valores podem ser -1=<FB<=1.  
Caso, após uma inserção qualquer valor da árvore fique com um fator de balanceamento diferente desses valores, a arvore se reestrutura mudando suas ligações para que todos os seu nós tenhamos esse fator de balanceamento. Sendo assim o tempo de busca assintótico ficará em torno de O independente da ordem de inserção dos valores.

3- criei um script python de inserção e balanceamento das chaves propostas no exercício, segue na pasta com o nome ‘3q.py’.

Teoria -> imagem completa, no script tentei imprimir no console, a formatação não está das melhores, mas o resultado bate. Imagem certa logo abaixo:

Diagrama, Esquemático

Descrição gerada automaticamente

**Árvore AVL** é uma [árvore binária de busca](https://pt.wikipedia.org/wiki/%C3%81rvore_bin%C3%A1ria_de_busca) balanceada, ou seja, uma árvore balanceada (árvore completa) são as árvores que minimizam o número de comparações efetuadas no pior caso para uma busca com chaves de probabilidades de ocorrências idênticas. Contudo, para garantir essa propriedade em aplicações dinâmicas, é preciso reconstruir a árvore para seu estado ideal a cada operação sobre seus nós (inclusão ou exclusão).  
  
**Inserção ->**

Para inserir um novo nó de valor K em uma árvore AVL é necessária uma busca por K nesta mesma árvore. Após a busca o local correto para a inserção do nó K será em uma subárvore vazia de uma folha da árvore. Depois de inserido o nó, a altura do nó pai e de todos os nós acima deve ser atualizada. Em seguida o algoritmo de rotação simples ou dupla deve ser acionado para o primeiro nó pai desregulado.

**Remoção ->**

O primeiro passo para remover uma chave K consiste em realizar uma busca binária a partir do nó raiz. Caso a busca encerre em uma subárvore vazia, então a chave não está na árvore e a remoção não pode ser realizada. Caso a busca encerre em um nó u o nó que contenha a chave então a remoção poderá ser realizada da seguinte forma:

Caso 1: O nó u é uma folha da árvore, apenas exclui-lo.

Caso 2: O nó u tem apenas uma subárvore, necessariamente composta de um nó folha, basta apontar o nó pai de u para a única subárvore e excluir o nó u.

Caso 3: O nó u tem duas subárvores: localizar o nó v predecessor ou sucessor de K, que sempre será um nó folha ou possuirá apenas uma subárvore; copiar a chave de v para o nó u; excluir o nó v a partir da respectiva subárvore de u.

O último passo consiste em verificar a desregulagem de todos nós a partir do pai do nó excluído até o nó raiz da árvore. Aplicar rotação simples ou dupla em cada nó desregulado.

Rotação

A operação básica em uma árvore AVL geralmente envolve os mesmos algoritmos de uma árvore de busca binária desbalanceada. A rotação na árvore AVL ocorre devido ao seu desbalanceamento, uma rotação simples ocorre quando um nó está desbalanceado e seu filho estiver no mesmo sentido da inclinação, formando uma linha reta. Uma rotação-dupla ocorre quando um nó estiver desbalanceado e seu filho estiver inclinado no sentido inverso ao pai, formando um "joelho".

Para garantirmos as propriedades da árvore AVL rotações devem ser feitas conforme necessário após operações de remoção ou inserção. Seja P o nó pai, FE o filho da esquerda de P e FD o filho da direita de P podemos definir 4 tipos diferentes de rotação:

**Rotação simples à direita**

Deve ser efetuada quando a diferença das alturas h dos filhos de P é igual a 2 e a diferença das alturas h dos filhos de FE é igual a 1. O nó FE deve tornar o novo pai e o nó P deve se tornar o filho da direita de FE. Segue pseudocódigo:

Seja Y o filho à esquerda de X

Torne o filho à direita de Y o filho à esquerda de X.

Torne X o filho à direita de Y

Rotação à esquerda

Deve ser efetuada quando a diferença das alturas h dos filhos de P é igual a -2 e a diferença das alturas h dos filhos de FD é igual a -1. O nó FD deve tornar o novo pai e o nó P deve se tornar o filho da esquerda de FD. Segue pseudocódigo:

Seja Y o filho à direita de X

Torne o filho à esquerda de Y o filho à direita de X.

Torne X filho à esquerda de Y

**Rotação dupla à direita**

Deve ser efetuada quando a diferença das alturas h dos filhos de P é igual a 2 e a diferença das alturas h dos filhos de FE é igual a -1. Nesse caso devemos aplicar uma rotação à esquerda no nó FE e, em seguida, uma rotação à direita no nó P.

Deve ser observado que as 3 possíveis combinações de alturas da subárvores T2 e T3 constam da figura (h, h-1; h, h; e h-1, h), implicando em nos respectivos balanços do nó u (0/0/-1) e do nó p (1/0/0).

**Rotação dupla à esquerda**

Deve ser efetuada quando a diferença das alturas h dos filhos de P é igual a -2 e a diferença das alturas h dos filhos de FD é igual a 1. Nesse caso devemos aplicar uma rotação à direita no nó FD e, em seguida, uma rotação à esquerda no nó P.

Cabe ressaltar, apesar de não estar relatado na literatura, que na remoção poderá ocorre a situação na qual u^.fb=0. Neste caso deverá ser aplicada rotação simples e o fator de balanço dos nós u e p serão respectivamente 1 e -1.

**Rotação à esquerda**

Caso o fator de balanço do nó p tenha valor +2, então haverá necessidade de aplicar rotação simples ou dupla à esquerda.

Para identificar se qual rotação aplicar, bastará analisar o fator de balanço do nó u, raiz da subárvore direita do nó p.

**Rotação simples à esquerda**

Caso o fator de balanço do nó u seja +1, então deverá ser aplicada uma rotação simples à esquerda. A figura a seguir mostra a configuração da árvore antes e depois da rotação.

Rotação dupla à esquerda

Caso o fator de balanço do nó u seja -1, então deverá ser aplicada uma rotação dupla à esquerda. A figura a seguir mostra a configuração da árvore antes e depois da rotação.

4-

Quando um elemento é removido de uma AVL, nós devemos garantir que a propriedade de balanceamento seja mantida. Depois de remover o elemento desejado, as subárvores devem ser rebalanceadas.

Passo 1, primeira remoção do nó 4.  
Desenho com traços pretos em fundo branco

Descrição gerada automaticamente

Necessária uma rotação a esquerda no nó pivot 6:

Diagrama, Esquemático

Descrição gerada automaticamente

Passo 2, segunda remoção do nó 8.  
Forma

Descrição gerada automaticamente

Necessária uma rotação a esquerda no nó pivot 12:

Diagrama, Esquemático

Descrição gerada automaticamente

Quando uma subárvore é rebalanceada devido a uma remoção dos seus nós, pode ser que os pais desse nó (pivot) se tornem desbalanceados. Esse efeito colateral pode ir subindo na árvore. Se isso ocorrer, esses nós deverão ser avaliados e rebalanceados, se necessário.