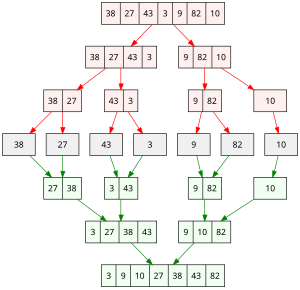
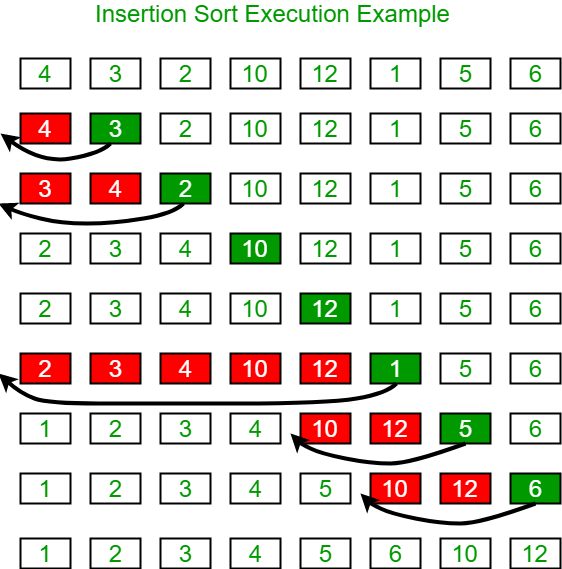
Pesquisa

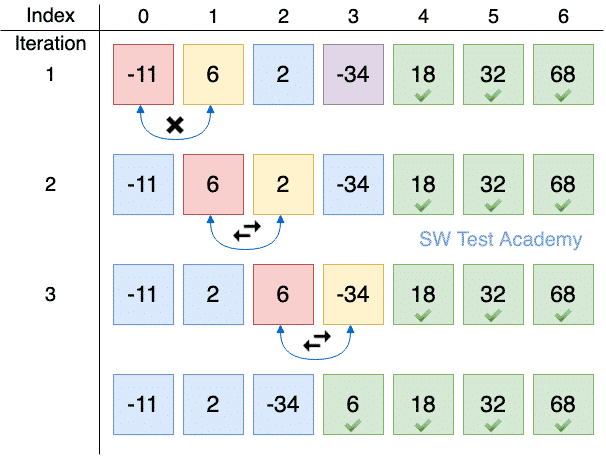
**Merge Sort -** É um algoritmo que cria uma sequência ordenada a partir de duas outras também. Ele divide a sequência original em pares de dados, agrupa estes pares na ordem desejada; depois as agrupa as sequências de pares já ordenados, formando uma nova sequência ordenada de quatro elementos, e assim por diante, até ter toda a sequência ordenada.



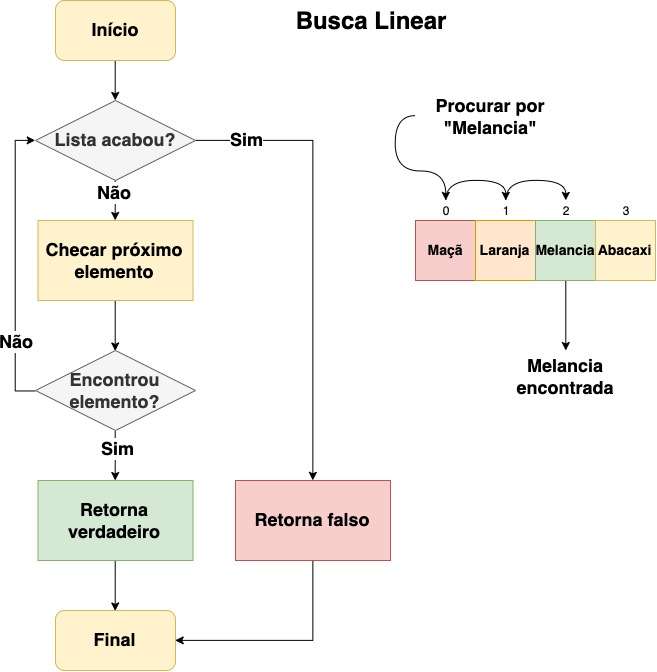
**Insertion Sort** -Ele inicia a ordenação dividindo o vetor em duas participações: uma ordenada à esquerda e outra não ordenada à direta. No início a partição esquerda só terá um elemento (é o caso trivial da ordenação). A inserção dos elementos na participação ordenada é realizada em duas etapas. Na primeira etapa, procuramos a posição de inserção. Na segunda etapa, deslocamos os elementos da esquerda para a direita para que a posição de inserção fique livre para que o elemento seja inserido.



**Bubble Sort** - É um algoritmo simples de ordenação que recebe como entrada uma lista de elementos e produz uma lista ordenada de acordo com um critério. O método funciona **examinando cada conjunto de elementos adjacentes na string, da esquerda para a direita, trocando suas posições se estiverem fora de ordem.**O algoritmo então repete esse processo até que possa percorrer toda a string e não encontrar dois elementos que precisem ser trocados.



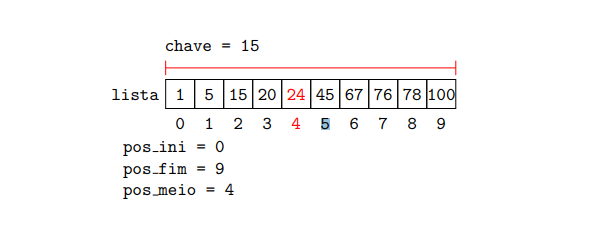
**Busca Linear –** É um algoritmo básico para encontrar um valor específico dentro de um arranjo ou lista de entrada, incluindo valores numéricos. Os algoritmos de busca linear envolvem a verificação sequencial de cada elemento na lista até que uma correspondência seja encontrada ou o final da lista seja alcançado. Essa pesquisa continua até que o valor desejado, seja numérico ou não, seja encontrado ou determinado que não existe na lista.

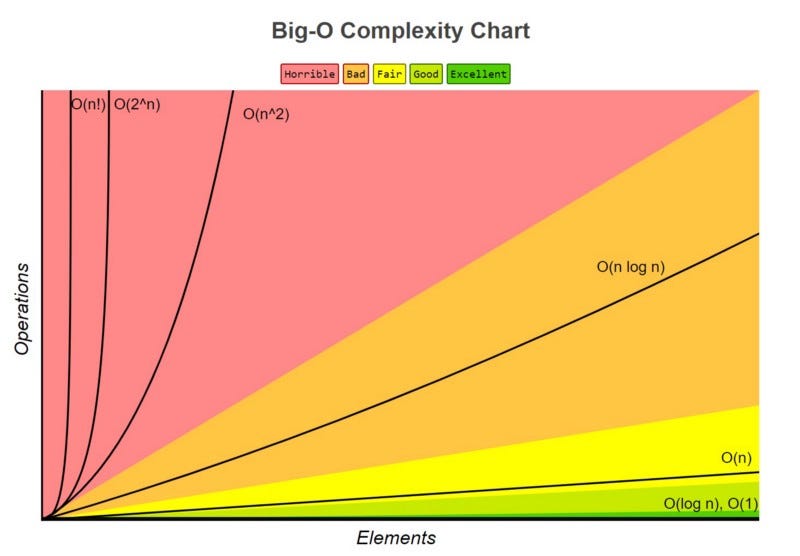


**Busca Binária** – Ele serve para encontrar um item em uma lista ordenada de itens. Ela funciona dividindo repetidamente pela metade a porção da lista que deve conter o item, até reduzir as localizações possíveis a apenas uma. Ele é muito mais rápido que a busca sequencial.

É um algoritmo eficiente, entretanto, requer que a lista esteja ordenada pelos valores da chave de busca, a ideia do algoritmo é a seguinte (assuma que a lista está ordenada pelos valores da chave de busca):

* Verifique se a chave de busca é igual ao valor da posição do meio da lista.
* Caso seja igual, devolva esta posição.
* Caso o valor desta posição seja maior que a chave, então repita o processo, mas considere uma lista reduzida, com os elementos do começo da lista até a posição anterior a do meio.
* Caso o valor desta posição seja menor que a chave, então repita o processo, mas considere uma lista reduzida, com os elementos da posição seguintes a do meio até o final da lista.



**Complexidade de algoritmo** - A complexidade de um algoritmo é analisada em termos de tempo e espaço. Normalmente, o algoritmo terá um desempenho diferente com a base no processador, disco, memória e outros parâmetros de hardware. A complexidade é usada para medir a velocidade de um algoritmo. Sendo o algoritmo um agrupamento de etapas para se executar uma tarefa, o tempo que leva para um algoritmo ser executado é baseado no número de passos.

Fontes:

<https://www.blogcyberini.com/2018/06/insertion-sort.html>

<https://elemarjr.com/clube-de-estudos/artigos/o-que-e-e-como-funciona-o-bubblesort/>

<https://www.cos.ufrj.br/~rfarias/cos121/aula_07.html>

<https://www.luigisbox.com.br/blog/tipos-de-algoritmos-de-pesquisa/#:~:text=Os%20algoritmos%20de%20busca%20linear,que%20n%C3%A3o%20existe%20na%20lista>.

<https://pt.khanacademy.org/computing/computer-science/algorithms/binary-search/a/binary-search#:~:text=A%20busca%20bin%C3%A1ria%20%C3%A9%20um,de%20adivinha%C3%A7%C3%A3o%20no%20tutorial%20introdut%C3%B3rio>.

<https://www.iugu.com/blog/analise-complexidade-algoritmos#:~:text=A%20complexidade%20%C3%A9%20usada%20para,baseado%20no%20n%C3%BAmero%20de%20passos>.