PESQUISA DE PWBE

Programação Web Backend

Kamila Vitória de Souza Cardozo

senai roberto mange  21/01/2025

**MERGE SORT**

Merger Sort é um algoritmo de classificação, ele segue um conceito de “dividir para conquistar”, a finalidade é criar uma classificação ordenada.

Passo a passo de como é feito a classificação por mesclagem:

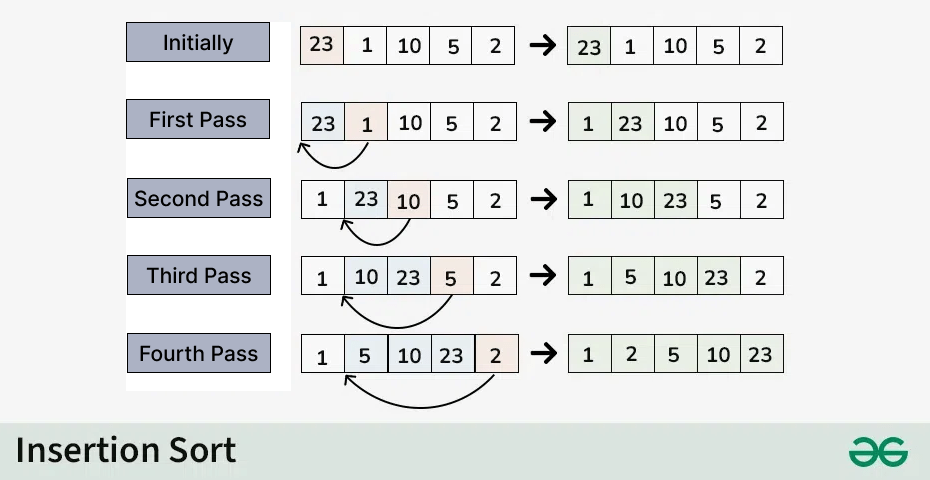
1. Dividir: o Merger Sort divide a lista ou a matriz várias vezes até que não seja mais possível realizar a divisão (até que os números fiquem em unidade);
2. Conquistar: Fazendo o uso do algoritmo de classificação por mesclagem, as submatrizes são classificadas de forma individual;
3. Mesclar: Após toda a classificação, os subarrays são mesclados novamente como estavam no inicio, mas dessa vez, todos em classificação ordenada, o processo é feito da mesma maneira como foi dividido, juntando aos poucos até que todos estejam mesclados novamente.

Observe o exemplo abaixo:



**INSERTION SORT**

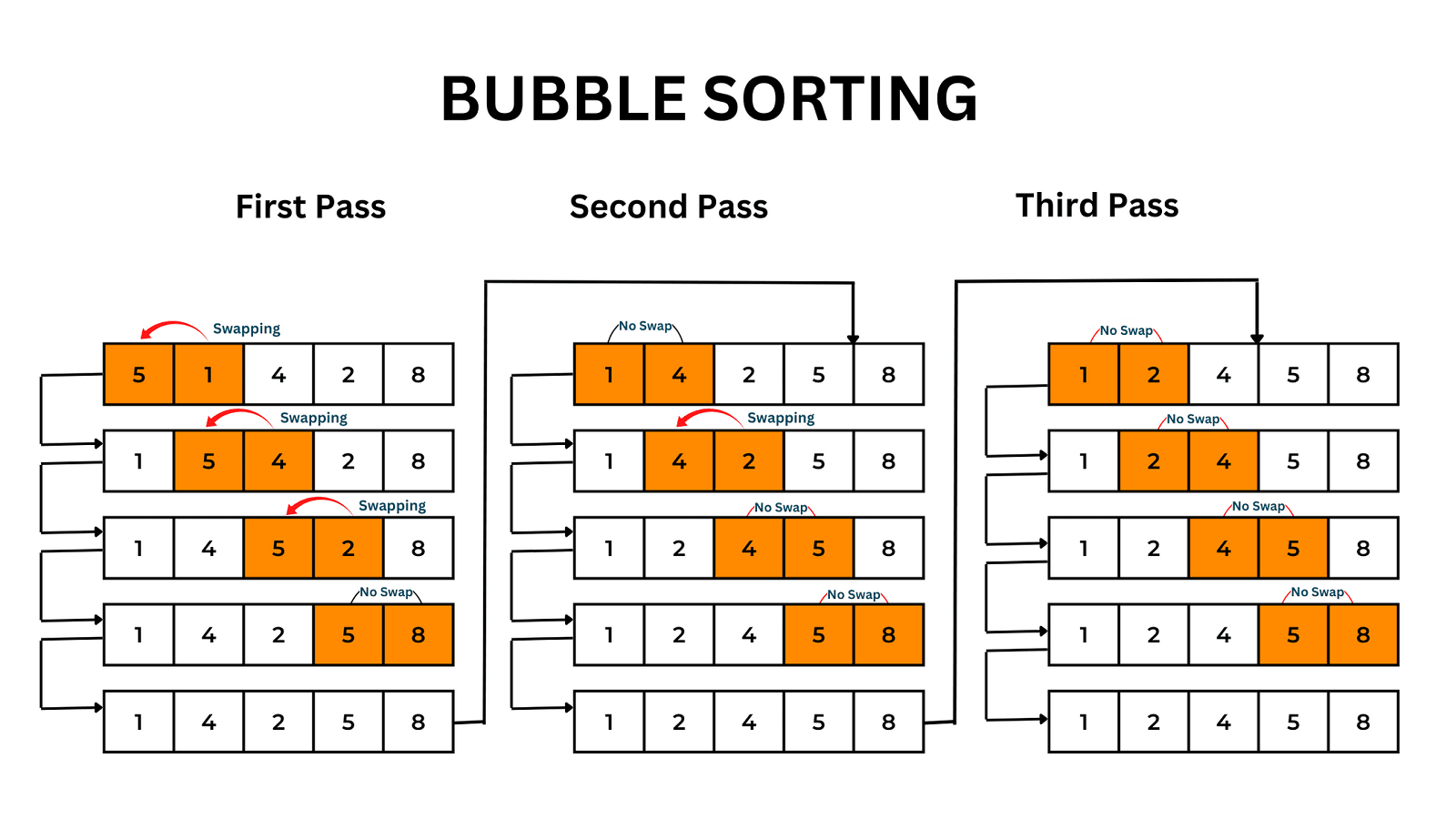
Insertion Sort é um algoritmo de classificação assim como Merge Sort, mas a sua diferença é que organiza cada elemento de uma lista não classificada sem a necessidade de dividir a lista, colocando cada elemento em seu devido lugar um por um com base no número que está ao seu lado.

* O segundo número é comparado com o primeiro e então verifica-se qual dos dois é o menor e o maior número, sendo assim, os números são trocados de posição se necessário, aonde o menor vai para frente e o maior para trás. Essa mesma troca acontece com os outros números, onde o terceiro número é comparado com os outros números que já foram selecionados, nesse caso o primeiro e o segundo e assim é colocado em seu devido lugar. A mesma coisa acontece com o quarto número e assim consecutivamente. 

**BUBBLE SORT**

Bubble Sort é um algoritmo de classificação, nesse método, os números são classificados adjacentemente, ou seja, um número é comparado com o número que se encontra ao seu lado dentro de uma lista, os números maiores ficam atrás e os menores na frente, esse processo se repetirá até que todos estejam em uma ordem ordenada correta.

O ponto negativo do método é a sua complexidade de tempo, pois como os números são analisados um a um de forma adjacente, ele se torna inviável para ordenar conjuntos que possuem muitos dados.



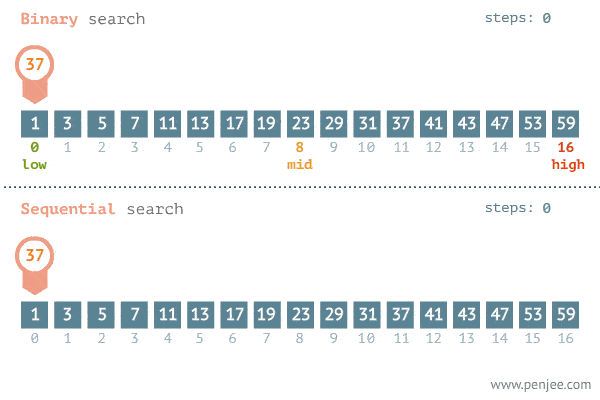
**O QUE É UM ALGORITMOS DE BUSCA**

Um algoritmo de pesquisa serve para recuperar os itens de qualquer estrutura de dados. Eles vão comparar os dados que são colocados na entrada com as informações que estão dentro do banco de dados e no fim, traz o resultado para o usuário. Um exemplo bacana é quando precisamos buscar um número de telefone específico dentro de uma lista com muitos números de telefones diferentes.

**ALGORITMOS DE BUSCA LINEAR**

O Algoritmo de Busca Linear desenvolve uma busca de forma sequencial, ou seja, ele percorre os itens a serem consultados um por um, do início ao fim.

Ele faz uma pesquisa percorrendo item a item afim de verificar cada um dos itens da lista e caso algum atenda as condições correspondes, esse item será retornado. Seu principal ponto positivo é que ele não exige que a lista seja ordenada para funcionar.



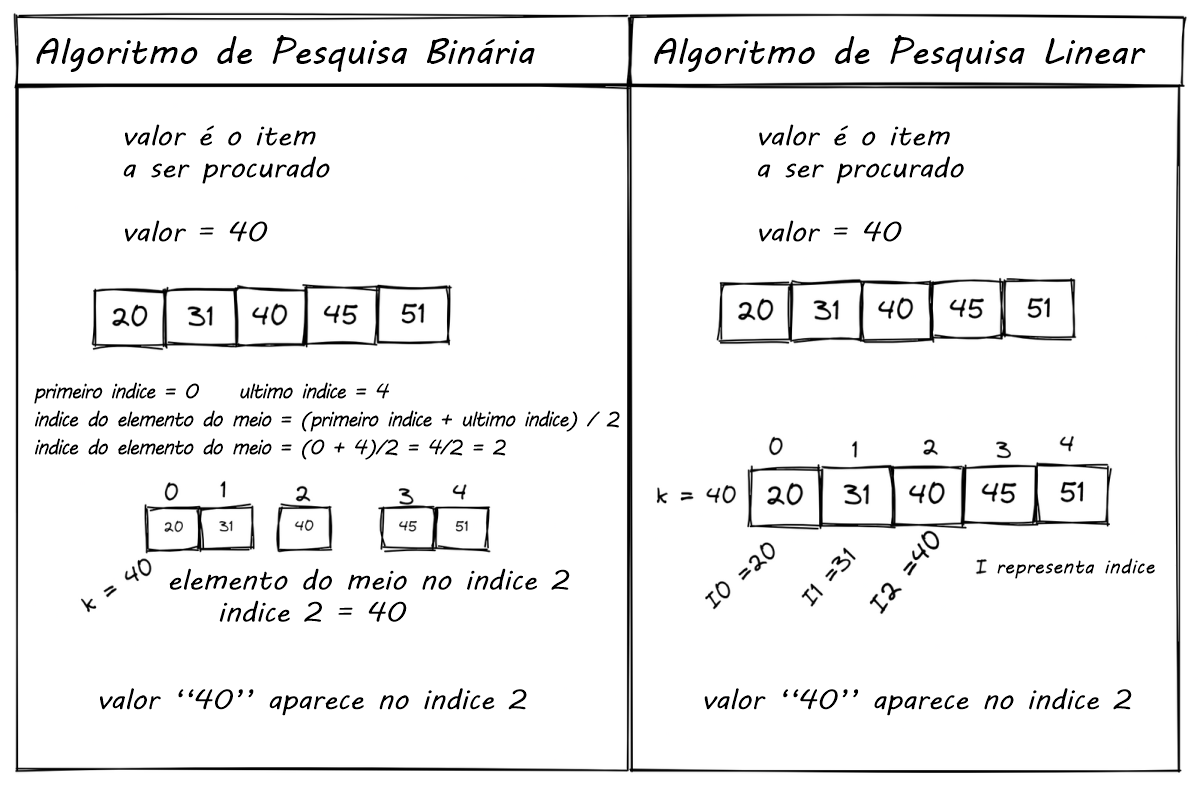
Ele vai analisar um por um até encontrar o item desejado que nesse caso do gif acima é o número 37.

**ALGORITMOS DE BUSCA BINÁRIA**

O Algoritmo de Busca Binária é conhecido como pesquisa de meio intervalo. Ele é um algoritmo de busca simples e segue o conceito de “Dividir para conquistar”. Para usar esse algoritmo é necessário que a lista seja uma lista classificada de forma ordenada.

Na busca binária, é definido o primeiro e o último índice, depois é feito a seguinte conta: (primeiro índice + segundo índice)/2, assim o resultado da conta que será um valor x é o índice que seria o meio da lista. A lista então é dividida em três partes, o lado esquerdo, o valor do meio e o lado direito,

Para utilizar o algoritmo de busca binária, é necessário que a lista seja ordenada de forma crescente, assim, quando a lista for dividida para encontrar o número do meio, os números a esquerda devem ser os menores valores e os da direita os maiores, afim de facilitar a busca do algoritmo.



**COMPLEXIDADE DE ALGORITMO**

A complexidade é analisada em relação a tempo e espaço. Normalmente, o algoritmo possui um desempenho diferente de acordo com o processador, o disco, a memória, entre outros fatores do hardware do computador onde será executado o processo. A complexidade tem o objetivo de medir a velocidade de um algoritmo.

O algoritmo é um conjunto de etapas que vão ser realizadas com a finalidade de executar uma tarefa. Por exemplo, um algoritmo percorre um array de dez posições, somando o índice das posições a 200. Sua complexidade seria de 10xˆt, sendo t = tempo (o tempo que foi necessário para conseguir atualizar cada um dos elementos do array com a ação da soma). Vale destacar que tudo irá depender do hardware do computador, ou seja, o tempo pode variar.

**CÁLCULO DA COMPLEXIDADE DOS ALGORITMOS ANTERIORES**

Para calcular o tempo de execução depende se é o pior caso ou no melhor caso:

**MELHOR CASO:**

No melhor caso, a lista já é ordenada, então o algoritmo não precisa ordená-lo. Sendo assim, o número de comparações é linear. A conta é feita da seguinte maneira:

Número de comparações: n -1, que é **O(n)**.

Número de trocas: **0**(zero) nenhuma troca é necessária pois a lista é ordenada.

Tempo de execução: no melhor caso **O(n)**.

**PIOR CASO:**

O pior caso acontece quando a lista não é ordenada, isso quer dizer que é necessário que o algoritmo faça as trocas dos números. A conta é O(nˆ2)

**GLOSSÁRIO**

**Array:** é uma estrutura de dados que armazena uma coleção de elementos do mesmo tipo, organizados de maneira na memória contígua (significa que a memória alocada para o array é um bloco contínuo). Esses elementos podem ser acessados de forma rápida e eficiente usando índices (comumente começam do zero), que são números inteiros que representam a posição de um elemento dentro do array.

**BIBLIOGRAFIA**

* https://www.geeksforgeeks.org/merge-sort/
* https://www.programiz.com/dsa/merge-sort
* https://www.geeksforgeeks.org/insertion-sort-algorithm/
* https://www.geeksforgeeks.org/bubble-sort-algorithm/
* https://favtutor.com/blogs/bubble-sort-python
* https://www.dio.me/articles/analise-de-algoritmos-tipos-de-busca-busca-linear
* https://www.freecodecamp.org/portuguese/news/pesquisa-binaria-em-python-como-escrever-o-algoritmo-de-pesquisa-binaria-e-exemplos/
* https://www.iugu.com/blog/analise-complexidade-algoritmos