Pesquisa de algoritmos

Merge Sort:

É um método que ordena utilizando o método dividir para conquistar. Ele divide uma lista em sub-listas, ordena as sub-listas e depois combina as sub-listas para uma lista final ordenada. Essas sub-listas tem só um elemento cada uma, depois que os elementos estão separados, eles são comparados e colocados à esquerda ou à direita de acordo com o valor. É bom para ordenar listas grandes.

Insertion Sort:

É um algoritmo simples e eficiente para ordenar elementos em uma lista com um pequeno número de elementos, ele compara um elemento chave com todos os elementos anteriores e muda sua posição na lista de acordo com a comparação de menor ou maior. É usado um loop do tipo for para percorrer a lista e um loop do tipo while para comparar elementos e mover eles de posição na lista.

Bubble Sort

É um algoritmo menos eficiente de ordenação. Ele percorre uma lista e compara os valores adjacentes e trocam eles de lugar de acordo com a comparação. É melhor ser utilizado em listas de tamanho pequeno ou se for executado em um computador com recursos de memória limitados. Ele tem a vantagem de ser mais fácil de entender e de realizar uma implementação do zero.

Busca Binária:

É um algoritmo de busca eficiente usado para encontrar um elemento específico em uma lista ordenada. Primeiro começa analisando se o item do meio é o que está sendo procurado, se for a busca termina, mas se não, como temos uma lista ordenada é possível eliminar uma metade da lista, se o elemento do meio for maior do o que está buscando é eliminado os elementos que vem

depois dele e a busca é feita na metade dos elementos que vem antes do meio analisado, isso serve para o contrário também. E essa metade da lista é de novo dividida até o elemento ser encontrado, ou a lista ficar vazia porque o elemento não está nela.

Busca Linear:

É um algoritmo que realiza a busca de forma sequencial, de forma a percorrer todos os itens da lista até encontrar o que está buscando. Sua eficácia depende da posição que elemento buscado está na lista, então isso a busca linear eficiente para listas com poucos elementos, pois sua busca é mais demorada. Ela também funciona para listas não ordenadas, como passa por todos os elementos.

Complexidade de Algoritmos:

Existem métricas para calcular a complexidade de um algoritmo, o que inclui tempo de execução, número de operações e taxa de crescimento. O tempo de execução se refere ao tempo total que o algoritmo leva para realizar todo o processo, o número de operações se refere a quantas operações primitivas que o algoritmo executa de acordo com o tamanho da entrada, a taxa de crescimento como o tempo de execução ou o úmero de operações cresce de acordo com o aumento da entrada. A notação Big O é uma forma padronizada de representação da complexidade do algoritmo.

- Complexidade do merge sort: O(nlogn) no pior, médio e melhor caso;
- Complexidade do insertion sort: $O(n^2)$, no pior e médio caso, já no melhor caso quando a lista já está ordena é O(n);
- Complexidade do bubble sort: $O(n^2)$, para o pior e médio caso, já no melhor caso é O(n);
- Complexidade da busca binária: O(log n), onde n é o tamanho da lista;
- Complexidade da busca linear: O(n), onde n representa o tamanho da lista;

Referências

https://www.dio.me/articles/analise-de-algoritmos-tipos-de-busca-busca-linear

https://napoleon.com.br/glossario/o-que-e-binary-search-2/

https://elemarjr.com/clube-de-estudos/artigos/o-que-e-e-como-funciona-o-bubblesort/#:~:text=Complexidade%20do%20algoritmo,complexidade%20%C3%A9%20O(n).

https://www.escoladnc.com.br/blog/a-importancia-da-complexidade-algoritmica-na-ciencia-da-computacao/

https://awari.com.br/busca-binaria-em-python-aprenda-a-encontrar-elementos-de-forma-eficiente/

https://awari.com.br/aprenda-a-implementar-o-algoritmo-de-merge-sort-em-python/

https://awari.com.br/aprenda-a-implementar-o-algoritmo-de-ordenacao-insertion-sort-em-python/

https://medium.com/rafaeltardivo/bubble-sort-passo-a-passo-67bb5e11677c