Estrutura de dados 2022.2
Professor : Thiago Sales

Nome: Jorge Lucas Firmino Silva de Sá
Número de matrícula: 22112410

Nome:Nicoli Valentim dos Santos
Número de matrícula: 22112017

Nome: Alex Sandro Antonio de Oliveira
Número de matrícula: 22112015

Banco de dados utilizado:
https://data.opendatasoft.com/explore/dataset/geonames-all-cities-with-a-population-1000%40public/

Trabalho sobre algoritmos de ordenação "Sorting"

Baseado nos estudos realizados em sala de aula e com a implementação dos algoritmos de ordenação "Quick Sort", "Merge Sort" e "Heap Sort" na linguagem de programação Python, o nosso grupo (Alex, Jorge, Nicoli) produziu um banco de dados com o nome de todas as cidades no mundo com uma população superior a 1000 ("mil") habitantes.
Para cada cidade foi relacionado um id para aplicação de cada "Sort" (Exemplo 1234578 : Nothingville) onde cada uma das cidades foi ordenada de forma crescente.

Breve explicação de cada algoritmo

Merge sort : Ele consiste em uma ordenação por divisão de array, onde é cedido uma lista para o mesmo, que realiza uma divisão deste array e estas partes são divididas e assim sucessivamente até cada array ser composto apenas por um item.
Realizada as divisões ele compara cada array único e ordena e assim procede para próxima camada com a finalidade de chegar ao ponto inicial com os itens já ordenados.

Quick sort : Com a escolha de um "pivô" o quick sort compara cada item do array com o mesmo para realizar a ordenação.
A cada comparação ele move o item menor do que o "pivô" para a esquerda e o maior para a direita realizando as correções a cada comparação.

Heap sort : Os dados são postos em uma árvore binária para realização de uma comparação da "folha com o pai", caso a folha for maior que o pai ela é trocada.
Para cada item comparado, ele pode ou não assumir uma nova posição.

Especificações de hardware

Nicoli:
Sistema operacional: Windows 10 Pro
Processador: Intel(R) Core(TM) i5 CPU M 430 @ 2.27GHz 2.27 GHz
Memória ram:8,00 GB
Disco de armazenamento: 465 GB

Alex:
Sistema operacional : Linux Mint 2 1.1 Cinnamon
Versão do cinnamon: 5.6.8
Kernel do linux : 5.15.0-70-generic
Processador: Intel© Celeron© CPU N2830 @ 2.16GHz ×2
Memória: 3.7 GiB
Discos rígidos: 241.1 GB
Placa de vídeo: Intel Corporation Atom Processador Z36xxx/Z37xxx Series Graphics & Display

Jorge:
Sistema Operacional:-Windows 10 Home Singles Language
Processador: Intel Core i3 7020U
Memória ram: 4GB RAM
Disco de armazenamento: HD 1TB
Placa de vídeo: Intel HD Graphics 620

Comparação de tempo de execução em cada máquina através do vscode

Nicoli:
Merge - 2.30s
Heap - 4.25s
Quick - 6.20s

Alex:
Merge - 2.21s
Heap - 3.99s
Quick - 6.00s

Jorge:
Tempo com 5000 objetos:
Merge: 0.95s
Heap: 1.71s
Quick: 2.62s

Tempo com 8000 objetos:
Merge: 0.89s
Heap: 1.95s
Quick: 2.56s

Em todos os casos pode se perceber que o algoritmo de ordenação Merge sort foi o mais eficiente em questão de tempo, enquanto o Quick sort necessitou de mais tempo para a execução.

Complexidade

Merge - O(n log n)
Heap - O(n log n)
Quick - Varia de O (n log n) para O(n^2) dependendo da escolha do pivô.