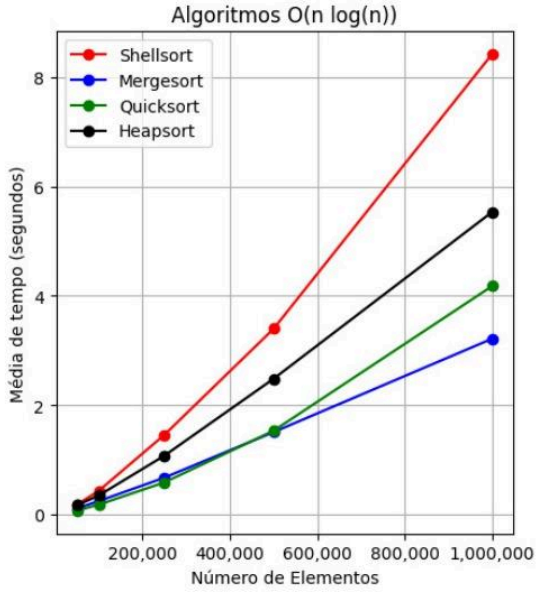
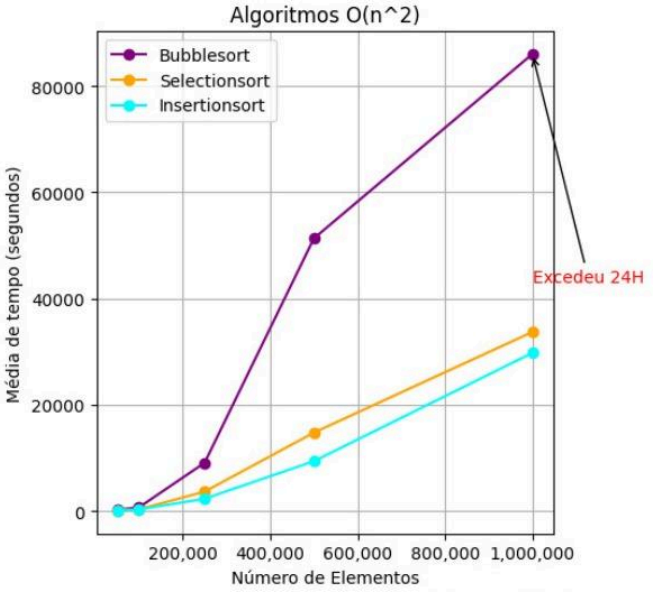
ALUNOS: Marcos Vinícius Duarte Sousa, Felipe Gabriel Soares Rodrigues, Cauã Martins da Silva.
PROFESSOR: Raimundo Moura.
CURSO: Ciência da computação - Bacharelado.
COMPONENTE CURRICULAR: Estruturas de dados.

RELATÓRIO TÉCNICO REFERENTE AO DESEMPENHO DOS ALG. DE ORDENAÇÃO

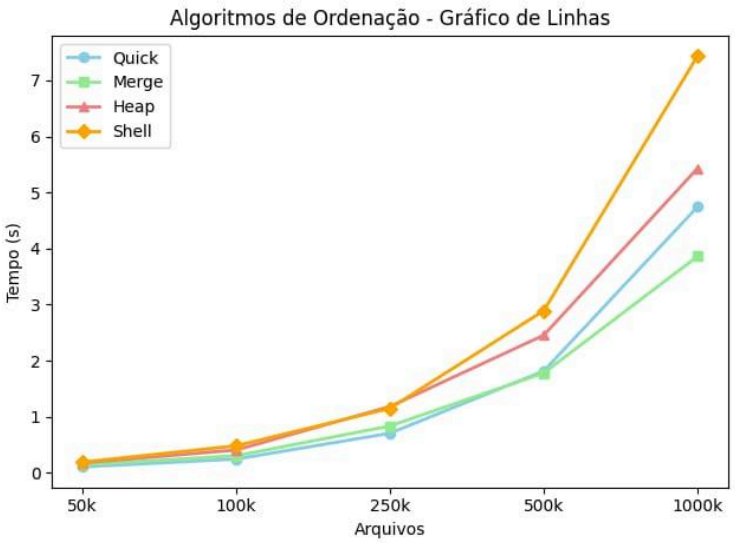
Algoritmos de ordenação usados: Bubble Sort, Selection Sort, Insertion Sort, Shell Sort, Merge Sort, Quick Sort, Heap Sort.

GRÁFICO DE DESEMPENHO:





Comparação com os resultados obtidos com outro grupo:
(Grupo 02 - Davi Sousa Soares, Victor Kauan da Silva Miranda, João pedro Saleh de Sousa)



Similaridades:

A ordem de velocidade dos algoritmos se manteve exatamente igual, com o Shell sendo o mais veloz e o Bubble o mais lento.

Diferenças:

Todos os nossos testes acabaram sendo um pouco mais lentos — talvez pela menor capacidade de processamento de nossos notebooks.

Tópicos importantes:

É notório como a complexidade dos algoritmos O(n log(n)) — Shell, Merge, Quick e Heap — torna seus desempenhos muito mais velozes do que os algoritmos quadráticos, e embora a complexidade entre os quatro seja igual, é possível perceber que a rapidez varia de caso para caso, dependendo da quantidade de elementos a ser ordenado e da forma como tais elementos estarão dispostos no vetor. Também vale ressaltar que há diversas formas de otimizar estes algoritmos, e que o grau de desempenho varia muito de versão para versão.

Já em relação aos quadráticos — Bubble, Insertion e Selection —, o Bubble é o que apresenta o maior tempo de todos (como a última checagem ultrapassou 24 horas, estimamos que esse tempo seja ainda maior). Essa lentidão deve-se ao fato de que o Bubble ordena por meio de comparações e trocas sucessivas, ao contrário do Insertion/Selection, que compara, mas faz apenas uma única troca ao final de cada iteração. Por esse motivo, o Bubble não é indicado para ordenação de uma sequência grande de dados, como é possível perceber com os casos iniciais (apresentados no início do gráfico), onde, por volta de 200,000 elementos, o tempo entre os três algoritmos é praticamente igual (o que muda à medida que a quantidade de elementos aumenta).

ESQUEMATIZAÇÃO:

Bubble sort:

Selection sort:

*Assim, a ausência de trocas consecutivas e "desnecessárias" torna os outros algoritmos mais rápidos.

Vale ressaltar, também, que há pouca diferença entre o Selection e o Insertion, embora este último tenha sido levemente mais rápido.

Assim, ao fazermos a média aritmética entre os resultados obtidos, chegamos a conclusão de que — pelo menos para os casos apresentados e para o desempenho obtido em nossos notebooks/PC's — o algoritmo mais rápido de todos foi o Shell Sort, e o mais lento o Bubble Sort.

PONTOS A CONSIDERAR SOBRE CADA ALGORITMO:

QUICK SORT: É rápido devido à sua capacidade de dividir e conquistar. Ele particiona a lista em duas sublistas menores, que são então ordenadas recursivamente. Seu desempenho é excelente em grande parte dos casos práticos. Além disso, ele não requer espaço extra significativo. (Criado por Tony Hoare em 1960)

MERGE SORT: É consistentemente eficiente com complexidade O(n log n) para todos os casos. Ele divide a lista em metades e depois mescla as metades ordenadas. É estável (mantém a ordem relativa dos elementos iguais) e eficiente para listas muito grandes. Utiliza espaço extra para a mesclagem. (criado por John von Neumann em 1945).

HEAP SORT: Usa um heap (monte) para organizar elementos. Ele constroi um heap máximo e então troca a raiz com o último elemento, repetindo o processo. Complexidade de tempo O(n log n) garantida para todos os casos. É in-place (não precisa de espaço adicional), mas pode ser mais lento na prática devido à manipulação do heap. (desenvolvido por J. W. J. Williams em 1964).

BUBBLE SORT: É simples de entender e implementar, mas ineficiente para listas grandes. (Criador não especificado).

SELECTION SORT: Menos eficiente, mas faz menos trocas em comparação com o Bubble Sort. (Criador não especificado).

INSERTION SORT: Eficiente para listas pequenas ou quase ordenadas, simples de implementar. (Criador não especificado).

SHELL SORT: É uma versão melhorada do Insertion Sort que supera a eficiência do Insertion Sort para listas grandes. É um algoritmo de ordenação que usa a técnica de "dividir e conquistar" através de gaps (intervalos). Em nossos testes, acabou sendo o mais rápido em todos os casos. (criado por Donald L. Shell em 1959).

FONTE: As versões de alguns dos algoritmos utilizados foram retiradas da plataforma digital "Geeks For Geeks", já outras foram implementadas pelo aluno Felipe Gabriel Soares Rodrigues.

https://www.geeksforgeeks.org/python-program-for-shellsort/
https://users.cs.duke.edu/~ola/bubble/bubble.html
https://www.geeksforgeeks.org/heap-sort/
https://www.geeksforgeeks.org/merge-sort/?ref=header_outind
https://www.geeksforgeeks.org/quick-sort-algorithm/?ref=header_outind