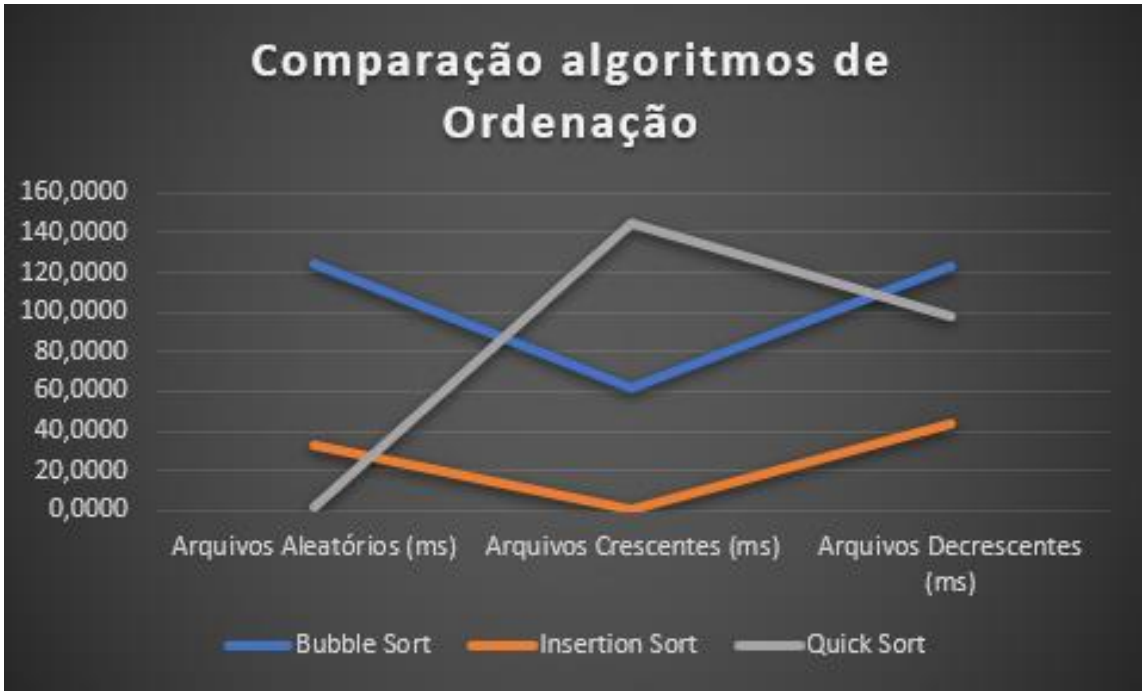
Relatório TDE 4 – Algoritmos de Ordenação

Integrantes: Claudio Colombo e Eric Simões

Este relatório apresenta uma análise comparativa de três algoritmos de ordenação clássicos — Bubble Sort, Insertion Sort e Quick Sort — em diferentes cenários e configurações de dados. O objetivo é avaliar o desempenho de cada algoritmo considerando três tipos de arranjos de entrada: dados aleatórios, dados ordenados em ordem crescente e dados ordenados em ordem decrescente, cada um com tamanhos variáveis (100, 1.000 e 10.000 elementos).

Resultados de Tempo de execução:

Algoritmo	**Arquivos Aleatórios (ms)**	**Arquivos Crescentes (ms)**	**Arquivos Decrescentes (ms)**
Bubble Sort	124.2349	62.0039	123.6167
Insertion Sort	33.2893	0.0350	43.2796
Quick Sort	1.5085	144.9558	97.8809



Dados Aleatórios

- **Algoritmo mais eficiente:** Quick Sort
- **Justificativa:** Quick Sort teve uma média de tempo significativamente menor (1.5085 ms) em relação aos outros algoritmos neste cenário. A razão para isso é que o Quick Sort, com complexidade média O(n log n), é bem adaptado a conjuntos de dados aleatórios, beneficiando-se do particionamento eficiente. Insertion Sort também apresentou bom desempenho para dados aleatórios, mas sua eficiência diminui à medida que o tamanho do conjunto aumenta, devido à sua complexidade O(n^2) O Bubble Sort foi o menos eficiente, com uma média de 124.2349 ms, pois sua natureza de troca repetida de elementos é mais lenta em conjuntos de dados de tamanhos maiores e com ordem aleatória.

Dados Ordenados Crescentes

- **Algoritmo mais eficiente:** Insertion Sort
- **Justificativa:** O Insertion Sort se destaca neste cenário, apresentando o menor tempo médio (0.0350 ms). Como o Insertion Sort é um algoritmo adaptativo, ele detecta que os dados já estão ordenados e reduz suas operações, quase alcançando O(n) em eficiência para esse tipo de dado. O Quick Sort, apesar de eficiente em casos médios, pode ter pior desempenho em dados já ordenados devido ao particionamento ineficaz, resultando em uma média de 144.9558 ms. O Bubble Sort, por outro lado, também se beneficiou da ordem crescente, mas ainda é limitado pela sua complexidade O(n^2).

Dados Ordenados Decrescentes

- **Algoritmo mais eficiente:** Quick Sort
- **Justificativa:** Para dados ordenados decrescentes, Quick Sort apresentou a melhor média (97.8809 ms), pois, mesmo sendo sensível à escolha do pivô, ele conseguiu particionar os dados de forma razoavelmente eficiente. O Insertion Sort foi menos eficiente (43.2796 ms) neste caso, mas ainda se saiu melhor que o Bubble Sort, pois só precisa fazer uma quantidade limitada de trocas. O Bubble Sort foi o mais lento, pois para dados decrescentes ele precisa executar o maior número de trocas possíveis, o que exacerba sua complexidade O(n^2).