UNIVERSIDADE EVANGÉLICA DE GOIÁS
BACHARELADO EM ENGENHARIA DE SOFTWARE

ÁRVORES E GRAFOS

ANDRESSA REZENDE SILVA - 2410138

Atividade Avaliativa – Implementação de Algoritmos de Ordenação em C

ANÁPOLIS –
2026

Implementação dos Algoritmos

Os algoritmos de ordenação foram implementados na ferramenta **VS Code**, utilizando a linguagem **C**. O **main** foi utilizado como corpo principal do programa. O conjunto de dados utilizado foi o mesmo para os cinco algoritmos, assim como o método de medição de tempo (utilizando clock). As únicas alterações realizadas foram, o código específico de cada algoritmo, a chamada da função no main, a medição de tempo individual para cada execução. Isso garantiu que todos fossem testados nas mesmas condições.

Resultados Obtidos nos Testes

Bubble Sort

- Quantidade de comparações: 49
- Quantidade de trocas: 20
- Tempo de execução: 42,00 ms

Selection Sort

- Quantidade de comparações: 28
- Quantidade de trocas: 7
- Tempo de execução: 36,00 ms

Insertion Sort

- Quantidade de comparações: 20
- Quantidade de trocas: 34
- Tempo de execução: 5,00 ms

Quick Sort

- Quantidade de comparações: 43
- Quantidade de trocas: 2
- Tempo de execução: 48,00 ms

Merge Sort

- Quantidade de comparações: 28
- Quantidade de trocas: 9
- Tempo de execução: 16,00 ms

Comparação Geral dos Algoritmos

Algoritmo	Como funciona	N comparações	N Trocas	Tempo de execução	Complexidade	Eficiência	Estável
Bubble Sort	Compara elementos adjacentes e troca se estiverem fora de ordem, repetindo até ordenar completamente.	49	20	42 ms	Média/Pior: $O(n^2)$Melhor: $O(n)$	Baixa para listas grandes	Sim
Selection Sort	Seleciona o menor elemento e coloca na posição correta a cada iteração.	28	7	36 ms	$O(n^2)$ em todos os casos	Baixa	Não
Insertion Sort	Insere cada elemento na posição correta dentro da parte já ordenada da lista.	20	34	5 ms	Pior: $O(n^2)$Melhor: $O(n)$	Alta para listas pequenas	Sim
Quick Sort	Escolhe um pivô, divide em menores e maiores e aplica o processo recursivamente.	43	2	48 ms	Médio/Melhor: $O(n \log n)$Pior: $O(n^2)$	Muito alta na prática	Não
Merge Sort	Divide a lista ao meio, ordena cada parte e depois intercala.	28	9	16 ms	$O(n \log n)$	Alta e previsível	Sim

Conclusão

Com base nos testes realizados:

- O **Insertion Sort** apresentou o menor tempo de execução (5 ms), sendo o mais eficiente para o conjunto de dados utilizado.
- O **Merge Sort** apresentou bom desempenho e estabilidade.
- O **Quick Sort**, apesar de ser considerado muito rápido na prática, apresentou maior tempo neste teste específico.
- O **Bubble Sort** e o **Selection Sort** apresentaram desempenho inferior, confirmando que são menos eficientes para conjuntos maiores.