

UNIVERSIDADE FEDERAL DE MINAS GERAIS
INSTITUTO DE CIÊNCIAS EXATAS
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO
ALGORITMOS E ESTRUTURAS DE DADOS II

Trabalho Prático 2

Algoritmos de Ordenação Interna

O objetivo deste trabalho é comparar o desempenho dos algoritmos de ordenação vistos em sala. Você deverá implementar os algoritmos **Bolha, Seleção, Inserção, ShellSort, QuickSort, HeapSort, MergeSort, RadixSort** e testá-los com diversos vetores de entrada (vetores de números inteiros), contabilizando o número de comparações de chaves, o número de movimentações de registros e o tempo de execução. Para isso você deverá colocar contadores em seu código e instrumentá-lo de forma a obter o tempo de execução. No caso do Quicksort, você deverá implementar a variação “mediana de três”, em que o pivô é escolhido usando a mediana entre a chave mais à esquerda, a chave mais à direita, e a chave central (como no algoritmo original).

Você deverá fazer tabelas comparando o desempenho de cada algoritmo. Mais especificamente, você deverá realizar testes com vetores de tamanhos 100, 1000, 10000 e 100000 elementos. Quatro diferentes tipos de vetores devem ser utilizados: **aleatórios, ordenados, quase ordenados (10% fora da ordem) e inversamente ordenados**. Para os vetores aleatórios, repita os testes pelo menos 10 vezes, de forma a obter médias do tempo de execução e dos contadores. O próprio programa deve gerar os vetores dos quatro tipos de entradas.

O seu programa deverá imprimir o método utilizado, o tipo e tamanho do vetor, o tempo de execução e o número de comparações e movimentações efetuado. Deverá haver também uma opção no programa para imprimir os vetores antes e depois da execução. Um detalhe importante é que o programa deverá ser executado passando-se opções na linha de comando. Esse tipo de execução é bastante comum em sistemas Unix / Linux e no antigo DOS. Por exemplo, os parâmetros do programa podem ser definidos assim:

ordena <algoritmo> <numero de itens> <situacao> [-P]

Onde

- **<algoritmo>** é um parâmetro que indica qual algoritmo será utilizado;
- **<numero de itens>** é um parâmetro numérico que indica quantas entradas devem existir no vetor;
- **<situacao>** indica como o vetor deve ser organizado antes de ser submetido ao algoritmo: aleatório, ordem crescente, quase ordenados (10% fora da ordem) e ordem decrescente;
- **[-P]** parâmetro opcional; caso presente, os vetores de entrada e saída devem ser impressos na tela.

Exemplos desse tipo de chamada seriam:

```
> ordena Ins 100 OrdC -P
> ordena Qui 100000 Ale
```

No primeiro caso, o programa executaria o procedimento **Inserção**, sobre um vetor ordenado de 100 elementos imprimindo os vetores antes e depois da ordenação. No segundo, seria executado o QuickSort com 100000 elementos aleatórios, sem a impressão dos vetores. Descubra como ler os parâmetros da linha de comando e defina (e documente) a sintaxe a ser utilizada. Os parâmetros de linha de comando podem ser especificados usando um janelado do DOS.

O que deve ser entregue

- Código fonte do programa em C ou C++ (bem indentada e comentada).
- Documentação do trabalho. Entre outras coisas, a documentação deve conter:
 1. Introdução: descrição do problema a ser resolvido e visão geral sobre o funcionamento do programa.
 2. Implementação: descrição sobre a implementação do programa. Deve ser detalhada a estrutura de dados utilizada (de preferência com diagramas ilustrativos), o funcionamento das principais funções e procedimentos utilizados, o formato de entrada e saída de dados, bem como decisões tomadas relativas aos casos e detalhes de especificação que porventura estejam omissos no enunciado.
 3. Estudo de Complexidade: estudo da complexidade do tempo de execução dos procedimentos implementados e do programa como um todo (notação O).
 4. Listagem de testes executados: os testes executados devem ser simplesmente apresentados.
 5. Conclusão: comentários gerais sobre o trabalho e as principais dificuldades encontradas em sua implementação.
 6. Bibliografia: bibliografia utilizada para o desenvolvimento do trabalho, incluindo sites da Internet se for o caso.
 7. A entrega DEVE ser feita pelo Moodle na forma de um único arquivo zipado, contendo o código, os arquivos e a documentação.

Obs: Consulte as dicas do Prof. Nívio Ziviani de como deve ser feita uma boa implementação e documentação de um trabalho prático:
<http://www.dcc.ufmg.br/~nivio/cursos/aed2/roteiro/>