Universidade Federal de Ouro Preto – UFOP
Instituto de Ciências Exatas e Biológicas – ICEB
Departam'ento de Computação – DECOM
Disciplina: Estruturas de Dados I – BCC202
Professor: David Menotti (menottid@gmail.com)
Professora: Emiliana Mara Lopes Simões (simoes.eml@gmail.com)

Trabalho Prático 5
Análise de Melhorias em Algoritmos de Ordenação por Comparação (Quicksort, Shellsort, Mergesort e Insertsort)

Valor: 0,5 pts (5% da nota total) **Data de entrega: 07/06/2010**
Documentação não-Latex: -0,1 pts
Impressão não frente-verso: -0,05 pts

Objetivos

Implementar modificações em algoritmos de ordenação por comparação clássicos e testar e analisar como essas modificações alteraram as comparações, atribuições e tempo de execução dos algoritmos.

Descrição

Este trabalho é composto por 3 sub-trabalhos práticos.Neste trablaho

1. Implementar o algoritmo Shellsort utilizando pelo menos 3 funções diferentes para gerar as h-partições. Lembre-se que a função *default* para este algoritmo é h(s) = 3.h(s)+1.

2. Implementar o algoritmo Quicksort utilizando pelo menos 4 formas diferentes de seleção do pivô.

3. **(Insertsort em pequenos vetores no Mergesort)** Sabe-se que o algoritmo Mergesort executa no pior caso em tempo O(n log n) e o algoritmo de ordenação por Insertsort no pior caso em tempo O(n^2). No entanto, os fatores constantes do Insertsort o tornam mais rápido que o Mergesort para um pequeno valor de n. Assim, faz sentido usar o Insertsort quando os sub-problemas tornam-se suficientemente pequenos. Considere a seguinte modificação no Mergesort: n/k sub-listas de comprimento k são ordenadas usando o Insertsort e então combinadas/intercaladas (*merged*) usando o mecanismo do Mergesort, sendo k o valor a ser determinado.
   a) Determine o valor de k através de experimentos.
   b) Compare o tempo de execução do algoritmo modificado com os algoritmos clássicos Mergesort e Inserção.

Como gerar os vetores/arranjos para ordenação

Considere vetores com a quantidade de elementos variando, *i.e.*, 10, 100, 1.000, 10.000, 100.000, 1.000.000, *etc*. Considere também vetores **sem valores repetidos**. Ainda, considere que todos os elementos dos vetores correspondem a valores inteiros e para gerar os vetores inicias, utilize vetores ordenados, inversamente ordenados, quase ordenados e aleatórios.

Universidade Federal de Ouro Preto – UFOP
Instituto de Ciências Exatas e Biológicas – ICEB
Departam'ento de Computação – DECOM
Disciplina: Estruturas de Dados I – BCC202
Professor: David Menotti (menottid@gmail.com)
Professora: Emiliana Mara Lopes Simões (simoes.eml@gmail.com)

O que analisar

A análise deve ser feita sobre o número de comparações, atribuições e tempo de execução dos algoritmos. Procure organizar inteligentemente os dados coletados em tabelas, e também construa gráficos a partir dos dados. Então, disserte sobre os dados nas tabelas e gráficos. Grande parte da avaliação será realizada sobre a análise dos resultados, ou seja, sobre o que você dissertar.

O que deve ser entregue

1. Código fonte do programa em C ou C++ (bem identado e comentado).
2. 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. **Muito importante**: os códigos utilizados nas implementações devem ser inseridos na documentação.
    3. Análise de Complexidade: estudo da complexidade de tempo e espaço das funções implementadas e do programa como um todo (notação O).
    4. Listagem de testes executados: os testes executados devem ser apresentados e analisados e discutidos, quando convier.
    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. Em Latex: Caso o trabalho não seja elaborado/escrito em latex, perde-se 0,1 pontos.
    8. Impressão: Caso o trabalho não seja impresso usando modo frente-verso, perde-se 0,05 pontos.
    9. Formato: mandatoriamente em PDF (http://www.pdf995.com/).

    Observação: Veja modelo de como fazer o trabalho em latex:
    http://www.decom.ufop.br/prof/menotti/aedI092/tps/modelo.zip

Como deve ser feita a entrega

A entrega DEVE ser feita via Moodle (www.decom.ufop.br/moodle) na forma de um **único** arquivo zipado, contendo o código, os arquivos e a documentação. Também deve ser entregue a documentação impressa na próxima aula teórica após a data de entrega do trabalho.

Comentários Gerais

1. Clareza, identação e comentários no programa também vão valer pontos;
2. Trabalhos copiados (e FONTE) terão nota zero;
3. O trabalho é individual (grupo de UM aluno);
4. Trabalhos entregues em atraso serão aceitos, todavia será descontado 0,1 pontos por teto(hora) de atraso;
5. Evite discussões inócuas com o professor em tentar postergar a data de entrega do referido trabalho.