LABORATÓRIO DE PROGRAMAÇÃO AVANÇADA

QUARTO TRABALHO PRÁTICO

COMPARAÇÃO DE MÉTODOS DE ORDENAÇÃO

OBJETIVO

Relembrar os conceitos de ordenação através da implementação de seis métodos e da comparação do desempenho entre eles.

PARTE 1

Deve ser enviado um relatório (**formato PDF**) contendo os dados abaixo.

		Aleatórios					
Métrica	**n**	**1K**	**5K**	**10K**	**20K**	**50K**	**100K**
Número de comparações	Bolha						
	Seleção						
	Inserção						
	Heapsort						
	Mergesort						
	Quicksort						
Número de trocas	Bolha						
	Seleção						
	Inserção						
	Heapsort						
	Mergesort						
	Quicksort						
Tempo	Bolha						
	Seleção						
	Inserção						
	Heapsort						
	Mergesort						
	Quicksort						

Como pode-se perceber, os seguintes métodos de ordenação devem ser implementados: Bolha, Inserção, Seleção, Heapsort, Mergesort, e Quicksort. A comparação entre os métodos de ordenação será feita através dos seguintes parâmetros: (a) número de comparações; (b) número de trocas; e (c) tempo de execução. Considere comparações feitas somente em **estruturas de seleção** que podem levar a uma troca, desconsiderando todas as outras.

As ordenações devem ser feitas em diferentes tamanhos de vetores, isto é, 1 mil, 5 mil, 10 mil, 20 mil, 50 mil e 100 mil. Estes vetores deverão ser gerados ALEATORIAMENTE. Usar a mesma massa de dados para TODOS os algoritmos de ordenação.

IMPORTANTE: Para minimizar possibilidades de que sejam gerados efeitos locais e, consequentemente, evitar viés no experimento, os dados que vão para o relatório acima são o VALOR MÉDIO de 10 (dez) execuções de cada algoritmo. Em cada uma das dez execuções, deve-se gerar novos números aleatórios. Não esqueça de usar a mesma massa de dados para TODOS os algoritmos de ordenação.

Resumindo: gera-se um vetor aleatório de **1000** elementos e aplica-se os algoritmos Bolha, Inserção, Seleção, Heapsort, Mergesort e Quicksort sobre esse vetor e armazena-se o número de comparações, o número de trocas e o tempo de execução de cada um dos **seis** algoritmos. Gera-se **novo vetor** aleatório de 1000 elementos e aplica-se novamente os seis algoritmos de ordenação e armazena-se novamente os três parâmetros de comparação. Repete-se mais **oito** vezes. Ao final, tira-se a **MÉDIA** do número de comparações, do número de trocas e do tempo de execução dos seis algoritmos. Preenche-se na tabela acima na coluna 1K.

OBSERVAÇÃO: Apesar de estar 1K, não significa 1024, mas sim 1 mil.

Repete-se o mesmo procedimento acima para as quantidades 5 mil, 10 mil, 20 mil, 50 mil e 100 mil.

PARTE 2

O objetivo é gerar três gráficos (**formato PDF**) das medições feitas, um para cada parâmetro: (a) número de comparações; (b) número de trocas; e (c) tempo de execução.

Para cada gráfico, deve-se incluir os dados dos seis métodos de ordenação, isto é, Bolha, Inserção, Seleção, Heapsort, Mergesort e Quicksort. Não esqueça de adicionar legendas para os métodos de ordenação.

Nos três gráficos, o eixo das abcissas (X) deve ser os tamanhos dos vetores: 1000, 5000, 10000, 20000, 50000 e 100000. No eixo das ordenadas (Y) deve ser um dos 3 parâmetros da comparação (comparações, trocas e tempo).

Resumindo, deverão ser entregues 3 gráficos, um para o número de comparações, um para o número de trocas, e outro para o tempo de execução. Sugiro que os gráficos sejam em linha.

OBSERVAÇÃO: como os dados do bolha devem ser bem maiores do que os dos outros, recomendo que, **se for o caso**, vocês coloquem o eixo dos Y (ordenadas) em ESACALA LOGARÍTMICA. Vi que o Excel, Calc do LibreOffice e GNU Plot tem essa opção. Vocês podem usar **qualquer software para fazer os gráficos.**

PARTE 3

Além da tabela e dos gráficos, incluir um relatório (**formato PDF**) contendo as respostas para as seguintes perguntas:

1. Considerando somente a métrica “tempo de execução” diga qual foi seu o melhor e o pior caso observado para cada método de ordenação?

   Por exemplo: Para o método da bolha o melhor caso foi para n=5 mil e o pior caso foi para n=100 mil. Para o método da inserção, o melhor caso foi . . .

2. Considerando somente a métrica “quantidade de comparações” diga qual função de n melhor descreve o desempenho de cada método de ordenação?

   Função aqui pode ser n, n^2, $\log_2 n$, $n.\log_2 n$, $n^2.\log_2 n$, $n^2 - n.\log_2 n$, $n^2 - (n.\log_2 n)^2$, etc. Vamos supor que os dados das quantidade de comparações do método da bolha para cada valor de n sejam: 959.639, 24.540.091, 98.711.127, 396.084.194, 2.487.915.240 e 9.951.660.960. Quando você compara com, por exemplo, n^2, dá os seguintes valores: 1.000.000, 25.000.000, 100.000.000, 400.000.000, 2.500.000.000, 10.000.000.000. Note que a quantidade de comparações se assemelha bastante com o valor de n^2. Repita o mesmo procedimento de busca para os outros métodos de ordenação considerando outras opções de funções. Sinta-se livre para propor a função que melhor se ajuste aos dados. Justifique através de dados numéricos o porquê da tua resposta.

3. Considerando as métricas “quantidade de trocas” e “quantidade de comparações” faça uma relação entre essas duas métricas e diga uma função que represente tal relação.

   Por exemplo, suponha que os dados de trocas sejam: 501.965, 12.480.626, 49.859.336, 200.026.552, 1.250.162.081; e os dados de comparações sejam: 959.639, 24.540.091, 98.711.127, 396.084.194 e 2.487.915.240. Pode-se concluir que a relação entre a quantidade de comparações (f(x)) e a quantidade de trocas (x) é: f(x) = x/2.

OBSERVAÇÕES GERAIS

- Se você jugar necessário, envie na parte 1 um arquivo README.txt para facilitar a reprodução do experimento.
- Todos os relatórios deve estar no formato PDF.
- Use somente as linguagens C ou C++.
- É obrigatório usar o sistema operacional Linux.
- Use a função **`gettimeofday()`** para calcular tempos, e **`srand()`** e **`rand()`** para gerar os números aleatórios.
- Mantenha sempre as mesmas unidades para comparação, por exemplo, tempo sempre em milissegundos ou segundos.
- Data de entrega: até o dia **13/04/2017 (quinta)** até 23:59. Após este prazo começa a contar o desconto progressivo, isto é, 0,1 ponto por hora de atraso.
- As três partes desse trabalho devem ser enviadas via **GIT HUB**. Será dado uma explicação na aula do dia 7 de abril de 2017.