BCC - Estruturas de Dados
Lista 08 - Ordenação

Prof. Dr. Paulo César Rodacki Gomes
IFC - Instituto Federal Catarinense

31 de maio de 2022

1 Objetivo

O objetivo desta atividade prática em laboratório é implementar e comparar o desempenho dos algoritmos BubbleSort, QuickSort e MergeSort. Escreva uma aplicação em C para ordenar arrays de números do tipo int.

Pede-se:

1. A aplicação deve implementar BubbleSort, QuickSort e MergeSort. Sugiro a criação de, pelo menos, uma função para cada algoritmo, de acordo com os protótipos abaixo. Neste caso, n é a quantidade de elementos do array, e v é o array de números inteiros a ser ordenado.

   - `void bubbleSort(int n, int* v);`
   - `void quickSort(int n, int* v);`
   - `void mergeSort(int n, int* v);`

2. Implemente uma função para criar dinamicamente vetores do tipo `int`, **com seus valores embaralhados**. A função deve usar algum recurso de randomização e ter o seguinte protótipo:

   `int* criaVetorEmbaralhado(int n);`, onde n é o número de elementos do vetor. Os valores armazenados no vetor nčo podem ser repetidos e devem ter valores de 1 a n (ou 0 a n-1 se você preferir). Note que o vetor é alocado dinamicamente, e deve ser usada uma funćčo de geração de números aleatórios para criar os elementos do vetor. Se o vetor tiver 10 elementos, os números devem ficar dentro do intervalo [1,10] (ou [0,9]). Se o vetor tiver 1.000 de elementos, os valores devem estar dentro do intervalo [1,1.000] (ou [0, 999]);

3. Implemente a função main, de forma a criar 3 vetores embaralhados de 10^1, 10^2, 10^3 e 10^4 elementos cada. Rode os três algoritmos com cada um dos vetores, registre o tempo gasto por cada um para encontrar a solução e imprima os resultados.