Oitava Lista de Exercícios
Busca e Ordenação

Norton Trevisan Roman

19 de maio de 2011

1. Escreva uma classe que guarde o código de um produto e o código de seu fabricante. Em seguida, escreva um método que ordene um aranjo de produtos (objetos dessa classe), passado como parâmetro, de acordo com o código de fabricante e, caso os produtos sejam de um mesmo fabricante, de acordo com o código do produto.

2. Escreva uma versão do método da bolha que ordena um arranjo de Strings

3. Escreva um método que ponha em ordem crescente uma seqüência desordenada de n números inteiros.

   (a) Usando ordenação por seleção

   (b) Usando ordenação por inserção

4. Suponha uma classe Cliente com os atributos abaixo:

   - int codigo;
   - String nome;
   - double salario;

   Suponha uma Classe CarteiraDeClientes com o atributo:

   - Cliente[] clientes;

   Implemente um método Cliente[] ordenaCodigoSelecaoDireta() que ordena o atributo clientes por código e devolve o arranjo ordenado (não alterando o atributo, ou seja, retorna um novo arranjo)

5. Considerando a mesma Classe CarteiraDeClientes do exercício anterior, escreva o método Cliente[] ordenaNomeInsercaoDireta(), que ordena o atributo clientes por nome e devolve o array ordenado (não alterando o atributo)

6. Algoritmos de ordenação são freqüentemente utilizados junto com outros algoritmos para torná-los mais eficientes.

   - Escreva uma classe que contenha um método atribuiArray que recebe um array de inteiros e o armazena como um atributo da classe. Escreva então um método int nRepetições(int x) que devolve o número de vezes que x aparece neste array
   - Escreva uma outra classe similar à anterior cujo método atribuiArray ordena os elementos do array antes de armazená-lo como atributo da classe.

Forneça a cada uma das classes vetores de diferentes tamanhos contendo números aleatórios. Para cada caso, chame o método int nRepetições(int x) utilizando diferentes valores de x. Podemos argumentar que a primeira classe é mais eficiente no caso em que são realizadas poucas consultas a nRepetições e a segunda classe é mais eficiente quando são realizadas muitas consultas. Justifique esta argumentação.

7. (Fusão de arranjos) Dadas duas seqüências ordenadas em ordem crescente números inteiros, passadas como parâmetro, escrever um método que retorne uma única seqüência ordenada a partir das seqüências originais.

8. Escreva um método que verifique se um dado valor x se encontra em uma seqüência em ordem crescente de n valores inteiros (use busca binária). Tanto x quanto a seqüência são dados como parâmetros. O método deve retornar true se o valor estiver lá e false se não.

9. Escreva um programa (classe com main) que descubra um número imaginado pelo usuário entre 0 e $n > 0$. Para cada valor sugerido pelo programa como sendo o valor imaginado pelo usuário, o usuário deve responder (honestamente) se o valor sugerido pelo programa é igual, menor ou maior do que o valor imaginado. A execução do programa deve terminar assim que o programa "adivinhar"o valor imaginado pelo usuário. (Dica: observe que isso pode ser resolvido com uma busca binária num arranjo de $n+1$ elementos, de 0 a n).

10. Seja uma matriz de 2 dimensões contendo números inteiros ordenados, de forma que cada linha esteja ordenada e que cada linha i contenha apenas elementos maiores ou iguais que os elementos da linha $i-1$, e menores ou iguais aos elementos da linha $i+1$. Ex:

$$\begin{pmatrix} -55 & -50 & -42 & -33 & -30 \\ -30 & -21 & -4 & 0 & 0 \\ 3 & 5 & 18 & 33 & 34 \\ 39 & 45 & 59 & 87 & 122 \end{pmatrix}$$

Escreva o método BuscaBinariaBidimensional que recebe como parâmetro uma matriz ordenada como descrita acima e um valor inteiro, e retorna a posição (i,j) onde esse valor ocorre na matriz (linha i, coluna j) ou (-1,-1) caso ele não esteja na matriz. Para essa busca, o método deve usar uma estratégia formada por 2 buscas binárias: uma para definir em qual linha buscar, e outra para buscar o valor pelas colunas dessa linha.