Exercício de Programação 1 – Listas Sequenciais

1) Escreva uma classe que mantém os dez maiores escores para uma aplicação de jogo (deve guardar nome do jogador e seu escore). Você deve utilizar lista seqüencial (implementação dada em aula com tipos genéricos – *ArrayIndexList*). Não é necessário que a lista seja ordenada.

Para tanto, você pode seguir os seguintes passos:

a) Você deve criar uma classe representando o escore de um jogador para um jogo. Essa classe deve ter como atributos o nome do jogador e seu escore no jogo. Você também deve, nesta classe, sobrescrever o método *toString()* (da classe *Object*).

b) (**Opcional**) Para comparação você pode implementar o método *compareTo(...)* da interface *Comparable* (mais detalhes abaixo). Outras formas de comparação serão aceitas e valerão a mesma nota.

c) Cada vez que um novo escore for adicionado na lista, você deve verificar se a lista já tem dez elementos (uma vez que só devem ser mantidos os 10 maiores escores). Caso a lista já tenha 10 elementos, você deve inserir o novo escore apenas se ele é maior que algum elemento da lista. Caso contrário, descarte-o.

d) Pense em como você irá modelar esse problema. Irá tratar tudo que é solicitado em um método *main()* de uma classe qualquer? Ou criará uma nova classe que contenha como atributo um objeto ArrayIndexList para poder adicionar outros métodos necessários? Existe ainda alguma outra possibilidade?

Sobre interface Comparable

Na Java Collection Framework, existem dois tipos abstratos de dados para verificar se um objeto é maior, igual ou inferior a um outro objeto: *java.lang.Comparable* e *java.util.Comparator*. Comparable é uma interface que deve ser implementada pela própria classe do objeto que se deseja comparar. Nesse caso, apenas uma ordem para esses objetos existem. Às vezes, se deseja implementar ordenações diferentes para a mesma classe. Neste caso você pode implementar para cada ordenação a interface *Comparator* em uma classe separada.

Comparable define o método *int compareTo (Object obj)* que retorna 0 se o objeto corrente for igual ao objeto *obj*, um valor negativo se *obj* é menor ou um valor positivo se *obj* for maior que o objeto para a qual este método foi invocado. A classe String implementa comparável, por exemplo.

Comparator define o método *int compareTo (Object o1, Object o2)*. Ele retorna 0 se o1 é igual a o2, um valor negativo se o1 for menor que o2, e um valor positivo se o1 é maior que o2.

Maiores detalhes em: http://download.oracle.com/javase/tutorial/collections/interfaces/order.html.

Exercício de Programação 2 – Pilhas

2) Escreva a classe Palindrome. Essa classe terá um único atributo (uma String) e os dois métodos descritos a seguir:
   a) Método inverse: Escreva o método inverse. Esse método usa uma pilha para inverter a ordem das letras de cada palavra de uma string, preservando a ordem das palavras. Por exemplo, dado o texto ESTE EXERCICIO E MUITO FACIL a saída deve ser ETSE OICICREXE E OTIUM LICAF. Ele retorna uma nova String que conterá o conteúdo do atributo string da classe invertido.
   b) Método isPalindrome: Escreva o método isPalindrome. Esse método retorna true se o atributo String é um palíndromo, false caso contrário. Use o método inverse para auxiliar nessa comparação.

O código deve estar dentro de um pacote Java chamado "*NomeSobrenomeExerc2*". Compactar apenas os arquivos fontes (com extensão *.java*).

Não se esqueça de colocar comentários com seu nome em cada arquivo de código fonte.