SD – Anselmo/Zizza - De Marco **Anno Acc. 2011-2012**

Appello riservato - 12 Novembre 2012

Cognome: **Nome:**

Matricola:

Attenzione:
Si consegna solo se la soluzione compila perfettamente e l'esecuzione rispetta i test forniti.

1. Una sequenza M di elementi si dice che è un anagramma di un'altra sequenza di elementi L se contiene gli stessi elementi lo stesso numero di volte (ossia è una permutazione delle stesse lettere).

   Usare l'ADT Mappa per scrivere un metodo **static <E> boolean isAnagram(PositionList<E> L, PositionList<E> M)** che restituisce true o false a seconda che la sequenza di elementi conservati in M sia un anagramma della sequenza di elementi conservati in L, oppure no. Ad esempio, se $L = \{1, 1, 4, 1, 4\}$ e $M = \{4, 1, 1, 1, 4\}$ il metodo deve restituire true; se $M = \{1, 1, 4, 1, 1\}$ oppure $M = \{5, 1, 1, 4, 1\}$ deve restituire false.

   Attenzione: soluzioni che non sono basate su un uso efficiente dell'ADT Mappa non saranno valutate.

   Questo esercizio va svolto all'interno della classe di test del pacchetto `nodelist`*. La correttezza della propria soluzione può essere testata all'interno della stessa classe.*

2. Si implementi con una lista di nodi un ADT BTContainer che sia un contenitore di alberi binari. Il contenitore dovrà avere, oltre a `size()` e `isEmpty()`, i seguenti metodi:

   (a) `BinaryTree<E> insert(E elem):` crea un albero binario avente l'elemento `elem` nella radice e lo inserisce nel contenitore.

   (b) `Position<E> insertLeft(BinaryTree<E> T, Position<E> v, E elem ):` aggiunge un figlio sinistro, avente `elem` come elemento, al nodo `v` dell'albero binario T.

   (c) `Position<E> insertRight(BinaryTree<E> T, Position<E> v, E elem ):` aggiunge un figlio destro, avente `elem` come elemento, al nodo `v` dell'albero binario T.

   (d) `BinaryTree<E> remove():` rimuove l'albero binario con il maggior numero di nodi interni.

   (e) `BinaryTree<E> remove(BinaryTree<E> T):` rimuove l'albero binario T dal contenitore. Lancia un'eccezione nel caso `T` non sia nel contenitore.

   Attenzione: i metodi (d) e (e) dovranno avere complessità $O(1)$.

   Questo esercizio va svolto all'interno del pacchetto ***`btcontainer`*** *che contiene anche l'interfaccia dell'ADT. La correttezza della propria soluzione può essere testata utilizzando il file di test presente nello stesso pacchetto.*