TP n°3 - Correction

Algorithmes de Tri (simples)

Exercices 1 et 2 : *Tris par sélection et insertion*

On considère un tableau t de taille n.
Pour le tri par sélection, on compare le premier élement aux n-1 autres élements du tableau. On compare ensuite le second élement aux n-2 autres élements et ainsi de suite jusqu'au bout du tableau. Le nombre de comparaisons est donc égal à : n-1 + n-2 +...+ 1. On a donc $Nb_{comp}(t) = \sum_{i=1}^{n-1} i$

A partir de là, 3 solutions :

1. On sait que $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$ donc, on a $Nb_{comp}(t) = \frac{n-1(n)}{2}$
2. On ne sait pas mais on a une idée : $2 * Nb_{comp}(t) = (n-1+n-2+...+1)+(1+2+...+n-1)$ d'où, par simplification, $2 * Nb_{comp}(t) = (n-1) * n$ et donc $Nb_{comp}(t) = \frac{(n-1)n}{2}$
3. On ne sait pas mais on a une autre idée sous forme de $\sum$:

   $\sum_{i=1}^{n-1} i = 1 + 2 + .. + n - 1 = n - 1 + n - 2 + ... + 1 = \sum_{i=1}^{n-1} (n - i)$

   d'où, $2 * Nb_{comp}(t) = \sum_{i=1}^{n-1} i + \sum_{i=1}^{n-1} (n - i) = \sum_{i=1}^{n-1} (i + n - i)$

   $2 * Nb_{comp}(t) = \sum_{i=1}^{n-1} n * 1 = n * \sum_{i=1}^{n-1} 1 = n * (n - 1)$ soit $Nb_{comp}(t) = \frac{n*(n-1)}{2}$

Dans tous les cas, on obtient une complexité en $O(n^2)$

La classe `TrisTableau` avec les méthodes de tri par sélection et par insertion :

```
class TrisTableau{

  public static void echange(int[] t, int p1, int p2){
    int temp=t[p1]; t[p1] = t[p2]; t[p2] = temp;
  }

  public static String TabtoString(int[] t){
    String s="";
    int i;
    for(i=0;i<t.length;i++){
        s += t[i]+"\t";
    }
    return s;
```

```
  }

  public static void triSelection(int[] t){
    int iMin;
    for(int iDebut=0;iDebut<t.length-1;iDebut++){
      iMin=iDebut;
      for(int i=iDebut+1;i<t.length;i++){
        if(t[iMin] > t[i])
          iMin=i;
      }
      echange(t,iDebut,iMin);
    }
  }

   public static void triInsertion(int[] t){
    int i;
    int elt;
    for(int iAInserer=0;iAInserer<t.length;iAInserer++){
      i=iAInserer;
      elt=t[iAInserer];

      /*tant qu'on est pas au debut du tableau et que l'element avant i est plus grand
      * que l'element à inserer, on decremente i et on décale les élements vers la droite
      */
      while((i>0)&&(elt < t[i-1])){
        t[i]=t[i-1];
        i--;
      }
      t[i] = elt; // insertion de l'element à la (bonne) place j
    }
  }
}
```

Un exemple de tests de ces méthodes :

```
public class Test{
  public static void main(String args[]){
    int[] tableau1={9,5,1,6,2};
    int[] tableau2={5,3,1,6,4,2};
    System.out.println(TrisTableau.tabToString(tableau1));
    TrisTableau.triSelection(tableau1);
    System.out.println(TrisTableau.tabToString(tableau1));
    System.out.println(TrisTableau.tabToString(tableau2));
    TrisTableau.triInsertion(tableau2);
    System.out.println(TrisTableau.tabToString(tableau2));
  }
}
```