TP MGL : Objects Calisthenics

MAAROUF ILIES

Les 9 règles à suivre pour un code “propre”:

1. Un niveau d’indentation par méthode
2. Le else est prohibé
3. Supprimer toutes primitives (int-> Integer) **??**
4. Une classe contenant une collection doit contenir uniquement les éléments de celle-ci.
5. Un point par appel methode (~~System.out.println~~)
6. Ne pas abbréger les noms de variables
7. Garder des classes inférieure à 50 lignes et des package inférieurs à 10 fichiers.
8. Pas de classe avec plus de deux types de variables instanciées
9. Pas de getter ni de setter la solution est de les remplacer par des méthodes perso et de leur donner des noms sémantiquement explicites en cas de besoin.

Exemple 1 : Tri à bulle

```
1
2  class  ApplicationTriBulle
3  {
4    static int [] table = new int[10];  // le tableau à trier en attribut
5
6    static void TriBulle  ( )    {
7      int n = table.length-1;
8      for (int i = n; i >= 1; i--)
9        for (int j = 2; j <= i ;j ++)
10         if (table[j - 1] > table[j]){
11          int temp = table[j-1];
12          table[j-1] = table[j];
13          table[j] = temp;
14        }
15     }
```

On observe les deux problèmes suivants :
* la primitive int que l’on remplace par Integer
*Et l’imbrication de deux for et des if

La solution est donée ici

```
class  ApplicationTriBulle_Calisthenics
{
  static Integer [] table = new Integer[10];  // le tableau à trier en attribut
  static Integer i;

  static void TriBulle() {
    Integer n = table.length-1;

    for (i = n; i >= 1; i--)
      TestElements(table);
  }

  private static void TestElements(Integer [] table){
    for (Integer j = 2; j <= i ;j ++)
      TestElement(table,j);
  }

  private static void TestElement(Integer [] table, Integer index){
    if (table[index - 1] > table[index]){
     Integer temp = table [index-1];
     table[index-1] = table[index];
     table[index] = temp;
   }
```

On remarque aussi que la méthode Impression contient System.out.println qui est contraire à la règle 5.

```
  static void Impression() {
    // Affichage du tableau
    int n = table.length-1;
    //Logger l = logger.getLogger("print");
   for (int i = 1; i <= n; i ++)
     System.out.print (table[i] + " , ");
    System.out.println ();
  }
```

On résoud ce problème avec un logger qui nous permet d’avoir un affichage tout en respectant le format calisthénique.

```
  static Logger print = Logger.getLogger("print");
```

```
   static void Impression() {
     // Affichage du tableau
     Integer n = table.length-1;
    for (Integer i = 1; i <= n; i ++)
     print.info(table[i] + " , ");

   }
```