

Effacement, Généricité, et Collections

🔪 **Exercice 1 : Effacement** Quelles sont les erreurs commises dans le code ci-dessous :

```
1  public class MyClass<E>{
2      E val ;
3      E[] tab1 ;
4      E[] tab2 ;
5      public static E shared ;
6      public static int nbInst ;
7
8      void f(){
9          val = new E();
10         tab2 = tab1;
11         tab2 = new E[3];
12     }
13 }
```

🔪 **Exercice 2 : Arbre Binaire de recherche** Lors de la séance précédente, vous avez implémenté un arbre binaire de recherche où les valeurs sont des **int**. Quel changement proposez vous afin de pouvoir utiliser votre code de façon plus générale.

🔪 **Exercice 3 : Gérer plusieurs Box!** On reprend les classes Box et TextBox du premier TD. Si on voulait réaliser une interface graphique, il faudrait gérer une structure qui rassemble des instances de Box.

Avec une liste

Implementez les méthodes suivantes (respectez bien les signatures) dans une classe ListGI (Comme pour ArrayList ou LinkedList, ici, je choisis le nom ListGI pour dire que c'est une Graphical Interface basée sur une liste).

1. **public void** addBox(int tl_x, int tl_y, int br_x, int br_y) qui ajoute une Box dont les coordonnées du coin en haut à gauche est (tl_x, tl_y) et les coordonnées du point en bas à droite est (br_x, br_y).
2. **public void** display(int n) qui affiche les informations de la box dont l'identifiant est passé en paramètre. Si l'identifiant correspond à un Box qui n'existe pas, écrivez un message.
3. **public void** removeBox(int n) qui enlève la Box dont l'identifiant est n.

4. **public void** listAllTopToBottom() qui affiche toutes les Box par ordre de l'ordonnée du coin en haut à gauche.
5. **public void** listAll() qui affiche toutes les Box par ordre des identifiants.

Avec une Map

Une Map est une relation qui associe à une clé une valeur. On peut reprendre les questions précédentes en remplaçant la liste de Box par une Map qui va associer l'identifiant de la Box et la Box associée (si elle est présente!). Implementez les méthodes suivantes dans une classe MapGI (respectez bien les signatures).

1. **public void** addBox(int tl_x, int tl_y, int br_x, int br_y) qui ajoute une Box dont les coordonnées du coin en haut à gauche est (tl_x, tl_y) et les coordonnées du point en bas à droite est (br_x, br_y).
2. **public void** display(int n) qui affiche les informations de la box dont l'identifiant est passé en paramètre. Si l'identifiant correspond à un Box qui n'existe pas, affichez un message.
3. **public void** removeBox(int n) qui enlève la Box dont l'identifiant est n.
4. **public void** listAll() qui affiche toutes les Box par ordre des identifiants. Pour cela, il faut utiliser un type de Map adéquat.
5. **public void** listAllTopToBottom() qui affiche toutes les Box par ordre de l'ordonnée du coin en haut à gauche. Pour cette question, pensez que la méthode `Collection<V> values()` retourne toutes les valeurs contenues dans une Map.
6. **public void** cleanAbove(int y) qui enlève toute Box dont le coin en haut à gauche se trouve au dessus de l'ordonnée y passée en paramètre.