ED/TP Collections et Héritage

1 Utilisation de collections

Exercice 1 Faire tourner le programme ci-après (après avoir rempli les blancs et corrigé toutes les fautes).

```
1 import java.util.*;
2 class ProgArrayList {
3    public static void main(String args[]) {
4           Colle_ _ _ _<String> obj = new ArrayList<String>();
5           obj.Add("A");
6           obj.Add("B");
7           obj.Add("B");
8           obj.Add("B");
9           obj.Add("C");
10          System.out.println("Liste:"+obj);
11          obj.remove("C");
12          System.out.println("Liste:"+obj);
13          obj.remove(1);
14          System.out.println("Liste:"+obj);
15   }
16 }
```

Exercice 2 Remplacer `ArrayList` par `HashSet` à la ligne 4. Pouvez vous anticiper le résultat ?

Exercice 3 Est-il possible de remplacer la Ligne 4 par

```
List<String> obj = new HashSet<String>();
```

?

Exercice 4 Remplacer la Ligne 4 par :

```
List<String> obj = new ArrayList<String>();
```

et ajouter à la fin du programme les lignes suivantes :

```
for(int i=0;i<1000000;i++)
  obj.add(0,"Last");
```

Le programme résultant, nécessite il plus d'une minute ?

Exercice 5 Remplacer `ArrayList` par `LinkedList`. Vous observez la différence du temps de calcul ?

2 Création de classes par héritage

Exercice 1 Écrivez en Java ou Processing une classe `CompteMinute` avec un attribut `minute` et avec un constructeur sans argument qui permet d'initialiser `minute`= 0. Le minuteur possède une seule méthode d'en-tête "`void avancerUneMinute()`" qui incrémente la valeur de l'attribut `minute`. Lorsque cet attribut arrive à la valeur 60, on doit remplacer 60 par 0. La classe ne devrait pas dépasser une douzaine de lignes!!

Exercice 2 Écrivez une classe `Horloge` qui hérite `CompteMinute`. L'objectif est de modéliser une horloge qui indique une heure et une minute (pas de seconde). Un objet de classe `Horloge` est initialisé par un constructeur avec deux arguments : l'heure et la minute. Comme pour toute classe, la méthode `toString()` renvoie une chaîne de caractères qui caractérise l'objet (par exemple, `21h54`). La méthode `avancerUneMinute()` permet de avancer le temps d'une minute. **Attention :** Si on avance d'une minute l'heure `23h59`, on obtient d'une minute `0h0`. N'oubliez pas de se servir de l'attribut `minute` de la classe `CompteMinute`.

Une définition correcte de la classe devrait faire fonctionner le programme ci-dessous et afficher la sortie indiquée à droite.

```
 1 Horloge monHorloge = new Horloge(2,59);
 2 System.out.println("Il est:");
 3 System.out.println(monHorloge);
 4 for (int i=0;i<60;i++){
 5     monHorloge.avancerUneMinute();
 6 }
 7 System.out.println("Apres une heure, il est");
 8 System.out.println(monHorloge);
 9 for (int i=0;i<60*24;i++){
10     monHorloge.avancerUneMinute();
11 }
12 System.out.println("Apres un jour, il est:");
13 System.out.println(monHorloge);
```

```
Il est:
2h59
Apres une heure, il est
3h59
Après un jour, il est
3h59
```