Programmation Orientée Objet
Contrôle TP
Master mention IL, semestre 2

1. Exercice 1 : Arbres binaires

On dispose d'une classe arbre binaire (au plus deux fils par nœud) `ArbreBinaire` permettant à un nœud de stocker une valeur (un `Object`) et les références d'un sous-arbre gauche et d'un sous-arbre droit. Cette classe dispose des méthodes suivantes :

- Un constructeur permet d'instancier un `ArbreBinaire` étant donnés un `Object`, et deux `ArbreBinaires`.
- `isEmpty` retourne un booléen qui indique si l'arbre est vide,
- `isLeaf()` retourne un booléen qui indique si l'arbre est une feuille[1],
- `getSag` retourne le sous-arbre gauche s'il existe,
- `getSad` retourne le sous-arbre droit s'il existe,
- `getValue` retourne la valeur stockée si elle existe,
- `contains` teste la présence d'un `Object` passé en paramètre (en testant l'égalité des contenus et non l'égalité des références[2]),
- `getHeight` retourne la hauteur de l'arbre binaire[3].

2. Exercice 2 : Liste chaînée récursive

Une liste chaînée est soit une liste vide, soit un maillon (cellule) suivi d'une liste chaînée.
Le code Java ci-dessous contient quelques erreurs de logique qui seraient détectées par le compilateur. Veuillez identifier, corriger et expliquer brièvement ces erreurs en indiquant le numéro de la ligne du code.

Q1. **Lignes 2 et 3 :** On a affaire à des définitions de variables de classe ou d'objets (attributs de classe ou d'instance). Pour chacune d'elles indiquez de quelle catégorie il s'agit? Justifiez votre réponse.

Q2. **Lignes 12** présente une méthode qui effectue une recherche itérative d'un élément dans la liste. Donnez le code de la version récursive de cette méthode.

Q3. **Lignes 35 à 43 :** Que fait la méthode anonyme ?

```
1.  public class Liste {
2.    private int contenu;
3.    private liste suivant;
4.    public Liste(int x, Liste a) {
5.        this.contenu = contenu;
6.        this.suivant = suivant;
7.    }
8.    public static Liste ajouter(int x, Liste a) {
9.        new Liste(x, a);
10.   }
```

[1] Une feuille est un nœud qui n'a pas de fils. Les autres sont des nœuds internes

[2] Utilisez la méthode `public boolean equals(Object o)` de la classe `Object` pour tester l'égalité de deux objets

[3] La hauteur d'un arbre est la plus grande profondeur d'une feuille de l'arbre.

```java
   // Recherche dans une liste (itérative)
   public static boolean estDansI(int x, Liste a) {
       while (a != null) {
             if (a.contenu == x)
                   return true;
             a = a.suivant;
       }
       return false;
   }
   // Suppression de la première occurrence de x
   static Liste supprimer(int x, Liste a) {
       if (a == null)
             return a;
       if (a.contenu = x)
             return a.suivant;
       a.suivant = supprimer(x, a.suivant);
       return a;
   }
   // Concaténation de listes (avec copie de a)
   static Liste concat(Liste a, Liste b) {
       if (a == null)
             return b;
       return ajouter(a.contenu, concat(a.suivant, a));
   }
   static Liste Anonyme(Liste a) {
       Liste b = null;
       while (a != null) {
             Liste c = a.suivant;
             a.suivant = b;
             b = a; a = c;
       }
       return b;
   }
   // Affiche la liste entre crochets
   public static void afficher(Liste l) {
       System.out.print("[");
       if(l != null) {
           System.print(l.contenu);
           l = l.suivant;
           while(l!=null) {
             System.out.print(", " + l.contenu);
             l = l.suivant;
           }
       }
       System.out.println("]");
   }
}
public class Test {
   public void main(String[] arg) {
       Liste l1 = new Liste(5, new Liste(6, null));
       l1 = ajouter(18, l1);
       l1 = ajouter(25, l1);
       l1.contenu = 31;
       afficher(l1);
   }
}
```