Université de M'sila M'sila le 10/01/2017
Faculté des mathématiques et d'informatique
Département de l'informatique Année Universitaire : 2016/2017
2ième année Licence Module : ASD1

Nom :
Prénom :
Groupe :

Interrogation TD

Exercice 1 :

Considérant trois listes simplement chainées, dont chaque élément est un entier, les trois listes sont ordonnées par ordre croissant.

Ecrire une méthode **UnionOrdre** permettant de fusionner les trois listes dans une liste ordonnée par ordre croissant puis retourner la liste résultat.

Solution :

```
public static LinkedList<Integer> fusionner(LinkedList<Integer>
l1,LinkedList<Integer> l2,LinkedList<Integer> l3)
    {
        LinkedList<Integer> l4 = new LinkedList<>();
        ListIterator<Integer> it = l1.listIterator();

        while(it.hasNext())
        {
            l4.add(it.next());
        }
        ListIterator<Integer> itf = l4.listIterator();
        it = l2.listIterator();

        while(it.hasNext()&& itf.hasNext())
        {
             Integer m = it.next();
             Integer n = itf.next();
             while( m > n && itf.hasNext()) n=itf.next();
             if (m<=n){
                itf.previous();
                itf.add(m);
             }
             else itf.add(m);
        }
        while(it.hasNext())
            itf.add(it.next());


        itf = l4.listIterator();
        it = l3.listIterator();
        while(it.hasNext()&& itf.hasNext())
        {
             Integer m = it.next();
             Integer n = itf.next();
             while( m > n && itf.hasNext()) n=itf.next();
             if (m<=n){
                itf.previous();
                itf.add(m);
             }
             else itf.add(m);
        }
        while(it.hasNext())
           itf.add(it.next());

         return l4;
    }
```

Exercice 2 :

Écrire une classe générique **Quadruplet** permettant de manipuler des quadruplets d'objets de différents types, on la dotera :

- d'un constructeur à quatre arguments.
- des méthodes d'accès, permettant d'obtenir la référence de l'un des éléments du quadruplet,
- d'une méthode affiche affichant la valeur des éléments du triplet.

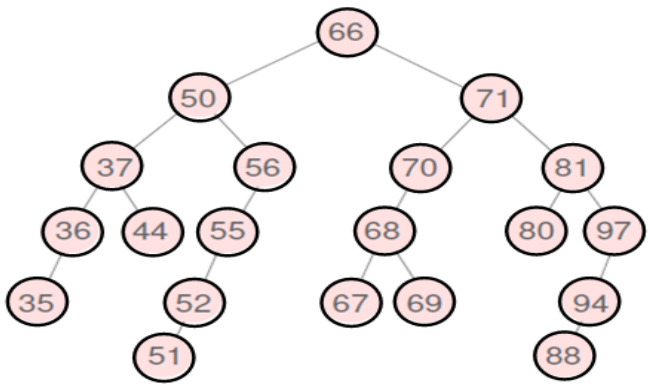
Écrire un petit programme utilisant cette classe générique pour instancier quelques objets et exploiter les méthodes existantes..

Solution :

```
public class Exo2 {

    public static void main(String[] args) {
        Quadruplet<Integer,Float,Double,String> q = new
Quadruplet<>(1,1.5f,3.5,"Quatre");
        q.afficher();

    }
}
class Quadruplet<T1,T2,T3,T4>
{
    private T1 x;
    private T2 y;
    private T3 z;
    private T4 t;
    public Quadruplet(T1 x,T2 y,T3 z,T4 t)
    {
        this.x=x;
        this.y=y;
        this.z=z;
        this.t=t;
    }
    public T1 prem(){return x;}
    public T2 sec(){return y;}
    public T3 troi(){return z;}
    public T4 quat(){return t;}
    public void afficher(){System.out.println("prem : "+x+" Sec : "+y+" Troi
: "+z+" Qaut : "+t);}

}
```

Exercice 3 :

Etant donné l'arbre binaire de recherche ABR suivant :



1. Afficher les nœuds de l'arbre ci-dessus en utilisant le parcours infixé ? que remarquez-vous ?
2. Afficher les nœuds de l'arbre ci-dessus en utilisant le parcours préfixé ?
3. Redessiner l'arbre ci-dessus après la suppression du nœud 71 ?

Solution :

1. Parcours infixé : 35 35 37 44 50 51 52 55 56 66 67 68 69 70 71 80 81 88 94 97
   Le parcours infixé donne les éléments d'un ABR selon l'ordre croissant.
2. Parcours préfixé : 66 50 37 36 35 44 56 55 52 51 71 70 68 67 69 81 80 97 94 88
3. L'arbre après suppression de 71 :