INSTITUT SUPERIEUR POLYTECHNIQUE DE MADAGASCAR

TP JAVA n°9 – POO et Graphes : Social Network

1) Implémentez les codes sources suivantes, dans un projet JAVA « Social » :

User.java

```
1   package social;
2   import java.util.List;
3   import java.util.ArrayList;
4   public class User {
5       private String name;
6       private List<User> followings;
7       private List<User> followers;
8       private List<Post> posts = new ArrayList<>();
9       public User(String name) {
10          this.name = name;
11          this.followings = new ArrayList<>();
12          this.followers  = new ArrayList<>();
13      }
14      public String getName() {
15          return name;
16      }
17      public void setName(String name) {
18          this.name = name;
19      }
20      public void follow(User u) {
21          if (u == null || followings.contains(u)) return;
22          followings.add(u);
23          u.followers.add(this);  // mise à jour du follower de u
24      }
25      public void createPost(String content) {
26          Post p = new Post(this, content);
27          posts.add(p);
28      }
29      public List<Post> getPosts() {
30          return new ArrayList<>(posts);
31      }
32      public void unfollow(User u) {
33          if (followings.remove(u)) {
34              u.followers.remove(this);
35          }
36      }
37      public List<User> getFollowings() {
38          return new ArrayList<>(followings);
39      }
40      public List<User> getFollowers() {
41          return new ArrayList<>(followers);
42      }
43      @Override
44      public String toString() {
45          return name + "\n  . Suivi(e)s : " + listToNames(followings)
46               + "\n  . Suiveurs   : " + listToNames(followers);
47      }
48      private String listToNames(List<User> list) {
49          if (list.isEmpty()) return "aucun";
50          StringBuilder sb = new StringBuilder();
51          for (User u : list) {
52              sb.append(u.name).append(", ");
53          }
54          sb.setLength(sb.length() - 2);
55          return sb.toString();
56      }
57  }
```

Post.java

```
1   package social;
2   import java.time.LocalDateTime;
3   import java.time.format.DateTimeFormatter;
```

```java
public class Post {
    private final User author;
    private final String content;
    private final LocalDateTime timestamp;
    public Post(User author, String content) {
        this.author    = author;
        this.content   = content;
        this.timestamp = LocalDateTime.now();
    }
    public User getAuthor() {
        return author;
    }
    public String getContent() {
        return content;
    }
    public LocalDateTime getTimestamp() {
        return timestamp;
    }
    @Override
    public String toString() {
        String time = timestamp.format(DateTimeFormatter.ofPattern("yyyy-MM-dd HH:mm"));
        return "[" + time + "] " + author.getName() + " : " + content;
    }
}
```

Main.java

```java
package social;
import java.util.List;
public class Main {
    public static void main(String[] args) {
        User alice   = new User("Alice");
        User bob     = new User("Bob");
        User charlie = new User("Charlie");
        User[] users = new User[]{alice, bob, charlie};

        alice.follow(bob);         // Alice suit Bob
        bob.follow(charlie);       // Bob suit Charlie
        charlie.follow(alice);     // Charlie suit Alice
        alice.unfollow(bob);       // Alice se désabonne de Bob
        charlie.follow(bob);       // Charile s'abonne à Bob
        System.out.println("\n=== Etat des relations ===");
        System.out.println(alice+"\n"+bob+"\n"+charlie);

        alice.createPost("Hello world!"); // création des posts
        alice.createPost("ISPM is the Best");
        bob.createPost("Hello INFO2 :)");
        charlie.createPost("Java is fun!");

        System.out.println("\n=== Posts de chaque utilisateur ===");
        for (User user : users) {
            for (Post p : user.getPosts()) {
                System.out.println(p);
            }
        }
        System.out.println("\n=== Timeline de Charlie ===");
        for (User following : charlie.getFollowings()) {
            for (Post post : following.getPosts()) {
                System.out.println(post);
            }
        }
    }
}
```

2) Dans la méthode `follow(User u)`, a-t-on besoin de faire une vérification supplémentaire à la ligne 23 ?
3) Dans la méthode `getFollowing()`, pourquoi on utilise `new ArrayList<>(following)` au lieu de retourner directement `following` ?
4) Expliquer les nuances entre `List<>` et `ArrayList<>`.
5) Dans la classe `User`, créer une méthode `getTimeline()` qui affiche les « posts des following » de l'utilisateur, similaire à ce qui a été fait dans `Main.java`, lignes 29…33.
   Est-ce que l'ordre d'apparition des `Posts` est optimal ?
   Proposer ou implémenter une solution.
6) Créer une classe `CommentedPost` qui hérite de la classe `Post`, avec la possibilité de faire des commentaires.