EST Salé
Département Informatique
LP BigData

Programmation Concurrente en Java
TP N°2 Thread

Exercice 1 Un problème d'accès concurrent

Voici 2 classes Compte (correspond à un compte bancaire) et Operation (thread qui effectue des opérations sur un compte bancaire).

```
public class Compte{
  private int solde = 0;

  public void ajouter(int somme) {
    //ajouter somme au solde actuel du compte
  }

  public void retirer(int somme) {
           //retirer, si c'est possible, la somme du solde actuel du compte
  }

 public void operationNulle(int somme) {
       //ajouter (int somme)
       //retirer(int somme)
  }

  public int getSolde() {
    return solde;
  }
}
```

```
public class Operation extends Thread {
       private String nomop;
       private Compte compte;

  public Operation(String nom, Compte compte) {
   this.nomop=nom;
    this.compte = compte;
  }

   public void run() {
    while (true) {
      int i = (int) (Math.random() * 10000);
      String nom = nomop;
      System.out.println("Nom opération  : "+nom);
      compte.ajouter(i);
      compte.retirer(i);
      compte.operationNulle(i);

      int solde = compte.getSolde();
      System.out.print(nom);
      if (solde != 0) {
         System.out.println(nom + ":*** Solde=" + solde);
       System.exit(1);
      }
    }
  }
```

1. Examinez le code et faites exécuter la classe Opération. Constatez le problème : opération effectue des opérations qui devraient laisser le solde du compte inchangé, et pourtant, après un moment, le solde ne reste pas à 0. Expliquez.
2. Modifiez le code pour empêcher ce problème.
3. Dans le code de Operation, remplacez l'opération nulle par 2 opérations ajouter et retirer qui devraient elles aussi laisser le solde du compte à 0 (elles sont en commentaire dans le code). Lancez l'exécution et constatez le problème. Modifiez le code pour que ça marche.

Exercice 2

Ecrire une classe qui représente deux threads ayant une chaine de caractères partagée. Les deux threads, tant qu'ils ne sont pas interrompus, alimentent cette chaine par une autre chaine.

Exercice 3 : synchronisation

Réaliser un programme illustrant deux threads qui alimentent un compte bancaire partagé.

Exercice 4 : Producteur/consommateur : ArrayList partagé

Supposons qu'on un tableau de données partagé par deux threads, l'un ajoute à la fin du tableau et l'autre supprime le dernier élément.
Réaliser un programme illustrant deux threads jouant les rôles de Producteur et consommateur.