TD de Compléments en Programmation Orientée Objet n° 5 : Multithreading : bases et primitives de synchronisation

Exercice 1 : Interférences

Quelles sont les sorties possibles de ce programme ?

```
1  public class Interferences {
2          static int x = 0;
3
4          static class ThreadIncr extends Thread {
5                  public void run() {
6                          for (int i = 0; i < 5; i++) {
7                                  System.out.println(++x);
8                                  try { sleep(1); }
9                                  catch (InterruptedException e) {
10                                         e.printStackTrace();
11                                 }
12                         }
13                 }
14         }
15
16         static class ThreadDecr extends Thread {
17                 public void run() {
18                         for (int i = 0; i < 5; i++) {
19                                 System.out.println(--x);
20                                 try { sleep(1); }
21                                 catch (InterruptedException e) {
22                                         e.printStackTrace();
23                                 }
24                         }
25                 }
26         }
27
28         public static void main(String[] args) {
29                 (new ThreadIncr()).start();
30                 (new ThreadDecr()).start();
31         }
32 }
```

Proposez des façons de corriger le programme.

Exercice 2 : Inconsistence

Quelles sont les sorties possibles de ce programme ?

```
1  public class Consistence {
2          static String s;
3          static int x = 0;
4
5          static class ThreadPrint extends Thread {
6                  @Override
7                  public void run() {
8                          System.out.println(x + " et " + s);
9                  }
10         }
11
```

```
12          public static void main(String[] args) throws InterruptedException {
13                  (new ThreadPrint()).start();
14                  (new ThreadPrint()).start();
15                  for (int i = 0; i < 10; i++) {
16                          s = "" + (x + 1);
17                          x++;
18                  }
19          }
20  }
```

Donnez plusieurs façon de s'assurer que le résultat sera celui auquel on s'attend.

Exercice 3 : Tri fusion

Soit le programme suivant, qui implémente le tri fusion de façon récursive, mais sur un seul thread :

```
1  public class TriFusion {
2    private static void triFusion(int tableau[], int deb,int fin) {
3      if (deb!=fin) {
4        int milieu=(fin+deb)/2;
5        triFusion(tableau,deb,milieu);
6        triFusion(tableau,milieu+1,fin);
7        fusion(tableau,deb,milieu, fin);
8      }
9    }
10
11   private static void fusion(int tableau[], int deb1,int fin1, int fin2) {
12     // le contenu de cette méthode ne nous intéresse pas : on sait juste
13     // qu'elle fusionne les deux sous-tableaux triés dans le bon ordre
14   }
15 }
```

1. Modifiez la méthode `triFusion()` pour qu'elle exécute chaque appel récursif dans un thread différent, puis opère la fusion seulement dès qu'elle est sûre que les sous-tableaux sont triés.
2. Remarquez que dans cet exemple, les appels récursifs modifient un même objet (le tableau). Comment pourrait-on faire pour récupérer les résultats intermédiaires, si ce n'était pas le cas (par exemple, si on travaillait soit sur un type primitif, soit sur des objets non mutables) ?
3. Remarquez aussi que dans notre exemple, deux appels concurrents à `triFusion()` modifient des parties disjointes du tableau. Qu'est-ce qui pourrait se passer si deux appels récursifs de même niveau ou susceptibles de s'exécuter en même temps, devaient modifier la même partie du tableau ?

Exercice 4 : Autres programmes récursifs

1. Concevez et écrivez un programme qui cherche la plus longue branche d'un arbre binaire en explorant récursivement les sous-arbres dans des nouveaux threads.

   On peut supposer qu'un arbre binaire est soit **null** (arbre vide), soit une racine avec deux sous arbres, c'est à dire un objet implémentant l'interface

```
1  interface ArbreBinaire {
2    ArbreBinaire ssArbreGauche();
3    ArbreBinaire ssAbrbreDroite();
4  }
```

2. Écrivez un programme qui calcule récursivement le n-ième terme de la suite de Fibonacci en lançant chaque appel récursif dans un nouveau thread. Est-ce que cette façon de faire est efficace ? Peut-on améliorer les choses, tout en continuant à profiter de la puissance d'un processeur multi-cœur ?

Exercice 5 : File d'attente

1. Dans la classe suivante, ajoutez des primitives de synchronisation afin que la lecture sur un tube vide soit bloquante (au lieu de retourner des résultats incohérents), de même que l'écriture sur un tube plein.

```
1  public class Tube {
2    private byte[] contenu;
3    private int iLecture;
4    private int iEcriture;
5
6    public Tube(int capacite) {
7      contenu = new byte[capacite];
8    }
9
10   byte lit() {
11     iLecture = (iLecture + 1) % contenu.length;
12     return contenu[iLecture];
13   }
14
15   void ecrit(byte b) {
16     iEcriture = (iEcriture + 1) % contenu.length;
17     contenu[iEcriture] = b;
18   }
19 }
```

2. On ajoute maintenant un producteur et un consommateur, qu'on exécute dans la méthode main, comme suit :

```
1          public static void main(String args[]) {
2                  Tube t = new Tube(2);
3                  (new Thread() {
4                          public void run() {
5                                  for (byte i = 0; i < 10; i++)
6                                          System.out.println("Lu : " + t.lit());
7                          }
8                  }).start();
9                  (new Thread() {
10                         public void run() {
11                                 for (byte i = 0; i < 10; i++) {
12                                         t.ecrit(i);
13                                         System.out.println("Écrit : " + i);
14                                 }
15                         }
16                 }).start();
17         }
```

Quelles sont les séquences d'affichage possibles pour ce programme ?