TP n°5

Les threads

Exercice 1 : Producteur/consommateur

On va s'intéresser ici au cas d'un producteur qui produit des objets, qui sont consommées par deux consommateurs différents. Ecrire les classes TabObject, Producteur et Consommateur, ainsi que la classe principale.

Exercice 2 : Un compteur

1) Un "compteur" a un nom (Toto par exemple) et il compte de 1 à n (nombre entier positif quelconque). Il marque une pause aléatoire (`Math.random()`) entre chaque nombre (de 0 à 5000 millisecondes par exemple).

Un compteur affiche chaque nombre (Toto affichera par exemple, "Toto : 3"), il affiche un message du type «**Toto a fini de compter jusqu'à 10**» à la fin.

Ecrivez la classe compteur et testez-la en lançant plusieurs compteurs qui comptent jusqu'à 10. Voyez celui qui a fini le plus vite.

2) Faites 2 versions : une où les threads sont créés avec une classe fille de Thread, et une où ils sont créés avec une instance d'une classe à part qui implémente Runnable.

3) Modifiez la classe Compteur pour que chaque compteur affiche son ordre d'arrivée : le message de fin est du type : «**Toto a fini de compter jusqu'à 10 en position 3**».

Exercice 3 : Tri parallèle

Voici un algorithme de tri en ordre croissant d'une tranche de tableau comprise entre les éléments d'indices debut et fin :

```
trier(debut, fin) {
  si (fin – debut < 2) {
    si (t[debut] > t[fin])
      echanger(t[debut], t[fin]) }
  sinon {
    milieu = (debut + fin) / 2
    trier(debut, milieu)
    trier(milieu + 1, fin)
    triFusion(debut,fin)//tri fusion des 2 moitiés debut à milieu et milieu + 1 à fin
  }
}
```

On remarque que les 2 tris qui sont effectués avant la fusion sont indépendants l'un de l'autre et il est donc facile des les faire exécuter en parallèle par 2 threads. A vous de jouer !

Vous pouvez vous servir du trieur monotâche fourni avec le TP (/TP5/tri).