Thread, interruption et variable de thread

Exercice 1 - Hello Thread

On souhaite créer deux threads exécutant le même code. Pour différencier les deux thread, lors de la construction de celle-ci, un entier unique (`id`) leur sera fourni, 0 pour la première et 1 pour la seconde.
Chaque thread executera le même code qui consiste à afficher `hello` suivi du numéro de la thread ansi que la valeur d'un compteur indiquant le nombre de fois que la thread a affiché ce message.
Exemple :

```
...
hello 0 10714
hello 0 10715
hello 0 10716
hello 0 10717
hello 1 15096
hello 1 15097
hello 1 15098
hello 1 15099
...
```

1 Expliquer sur l'exemple ci-dessus pourquoi le compteur de `hello 0` est beaucoup plus petit que le compteur de `hello 1`.

2 Écrire le programme demandé en héritant de la classe `Thread` et en redéfinissant sa méthode `run()`.

3 A quoi sert l'interface `Runnable` ?
Modifier votre code pour utiliser un `Runnable`.

4 Changer votre code pour que l'on puisse choisir lors de l'exécution du programme le nombre de thread que l'on veut lancer en concurrence.

```
java HelloThread 5
```

Lance 5 threads en concurrence.

Exercice 2 - Coitus interruptus

On souhaite maintenant permettre à l'utilisateur en tapant un nombre d'interrompre le thread ayant cet `id`.

1 Expliquer les différences entre la méthode `interrupted` et `isInterrupted` de la classe `Thread`.

2 Écrire la classe `Interruptus` en utilisant un scanner (`java.util.Scanner`) pour lire l'entrée standard et `interrupt()` pour interrompre un thread.

3 Monitoring de l'activté de la machine virtuelle.
Lancer votre programme avec la commande :

```
java -Dcom.sun.management.jmxremote HelloThread 5
```

Puis dans un autre shell lancer la commande :

```
jconsole
```

et sélectionner `HelloThread` dans les processus locaux. Vérifier par ce biais que les threads sont bien interrompus.

Exercice 3 - Modification d'une variable en concurrence

On souhaite créer deux threads qui change le même un champs d'un même objet :

```
public class Test {
      int value;

      public static void main(String[] args) {
        final Test test=new Test();

        for(int i=0;i<2;i++) {
          final int id=i;
          new Thread(new Runnable() {
            public void run() {
              for(;;) {
                test.value=id;
                if (test.value!=id)
                  System.out.println("id "+id+" "+test.value);
              }
            }
          }).start();
        }
      }
    }
```

1 Qu'affiche le code suivant avec la ligne de commande : `java Test` ?
Expliquer.

2 Qu'affiche le même programme mais avec la commande `java -server Test`
Expliquer.

3 Comment doit on faire pour être sûr que chaque thread voit les modifications effectués sur une variable par l'autre thread ?

Exercice 4 - strtok

La méthode `strtok` :

lorsqu'elle est appelée avec un chaîne de caractère (`CharSequence`) et un délimiteur (`char`), renvoie un `CharSequence` correspondant à la chaine du début de la chaine jusqu'à la prochaine occurence du délimiteur.

lorsqu'elle est appelée avec `null` et un délimiteur, renvoie un `CharSequence` de la position lors du dernier appel à la méthode jusqu'à la prochaine occurence du délimiteur.

Voici le code :

```
    public static CharSequence strtok(CharSequence input,char
delimiter) {
      int offset;
      if (input==null) {
        input=lastInput;
        if (input==null)
          return null;

        offset=lastOffset;
      }
```

```
      else {
        lastInput=input;
        offset=0;
      }

      for(int i=offset;i<input.length();i++) {
        if (input.charAt(i)==delimiter) {
          lastOffset=i+1;
          return input.subSequence(offset,i);
        }
      }
      lastInput=null;
      return input.subSequence(offset, input.length());
    }

    private static CharSequence lastInput;
    private static int lastOffset;
```

et un exemple :

```
     CharSequence seq1=strtok("toto est beau",' ');  // toto
     CharSequence seq2=strtok(null,' ');             // est
     CharSequence seq3=strtok(null,' ');             // beau
     CharSequence seq4=strtok(null,' ');             // null
```

1 Rappeler pourquoi la méthode `strtok` ci-dessous n'est pas thread-safe..

2 Faire en sorte que la méthode `strtok` soit thread-safe en utilisant des variables locales à un thread.