Prácticas Concurrencia y Distribución (22/23)

Arno Formella, Alba Nogueira Rodríguez, David Ruano Ordás

semana 6 – 10 febrero

Las prácticas de este curso están organizadas en actividades más o menos semanales para mantener un esfuerzo constante a lo largo del curso.

Práctica 1: Introducción a la concurrencia en Java

Objetivos: Adquirir conocimientos básicos sobre la forma como está implementada la concurrencia en Java.

Material adicional: Son de especial interés los siguientes enlaces (los de aquí son los más recientes, ojo, quizá el software en el laboratorio o en tu ordenador no es tan avanzado, en este caso usa la documentación adecuada para el sistema que usas).

- http://docs.oracle.com/javase/tutorial/essential/concurrency/index.html
- http://docs.oracle.com/javase/19/docs/
- https://docs.oracle.com/en/java/javase/19/docs/api/java.base/java/lang/Thread.html
- http://www.doxygen.nl/
- http://en.wikipedia.org/wiki/Markdown

Requisito general a todos los programas concurrentes que implementemos: El programa debe **terminar siempre** y todos sus subprocesos/hilos con un mensaje parecido a *Program of exercise X has terminated*, es decir, se garantiza que **todos los componentes** del programa concurrente terminan correctamente su ejecución.

Las preguntas que aparecen intercaladas en los enunciados tienen como objetivo animar a la reflexión y al auto-aprendizaje, servir como ejemplos de posibles preguntas en la fase de evaluación (examen), y ayudar a fundamentar los breves informes que se deben entregar en las actividades.

1. Examina en el manual y con ejemplos las dos formas que provee Java para crear un hilo: la clase `Thread` y la interfaz `Runnable`.

   ¿Hay alguna diferencia de funcionamiento entre ambas formas? ¿A nivel de diseño, cuál te parece preferible, y por qué?

2. Utiliza la forma con `Runnable` para implementar un programa que cree y ejecute tantos hilos como se le indica **via línea de comando**.

3. Extiende tu programa del apartado anterior para

   - que cada hilo imprima en pantalla un mensaje como *Hello world, I'm the java thread number X*.
   - y después de uno o varios segundos (usa otro argumento via línea de comando que indica el número de segundos), un mensaje *Bye from thread number X*,

   ¿Las salidas del programa reflejan lo que esperabas?

4. Extiende tu programa del apartado anterior para que en el método `main` de la clase principal se cree una lista (o un vector) de hilos e inícialos. Inmediatamente después de iniciar los hilos, desde el hilo principal, imprime a la pantalla un mensaje, p.ej., el programa ha terminado. Para crear una lista de hilos debes tener una variable `number_of_threads` que recoge el numero de hilos especificado por línea de comando:

   ```
   int number_of_threads;

   List<Thread> threadList =
     new ArrayList<Thread>(number_of_threads);

   for(int i=1; i<=number_of_threads; ++i) {
     //...
   }
   ```

   ¿Cuál es el resultado de tu código? ¿En qué orden se imprimen los hilos?