Programación Concurrente y de Tiempo Real*
Grado en Ingeniería Informática
Asignación de Prácticas Número 2

Se le plantean a continuación un conjunto de ejercicios sencillos de programación sobre el uso de la herencia en Java y de utilización de hebras concurrentes, que debe resolver de forma individual como complemento a la segunda tercera sesión práctica. Para cada uno, debe desarrollar un programa independiente que lo resuelva. **Documente todo su código con etiquetas (será sometido a análisis con javadoc)**. Si lo desea, puede también agrupar su código en un paquete de clases, aunque no es obligatorio.

1. Ejercicios

1. Utilizando herencia de la clase `Thread`, cree una condición de concurso[1] sobre una variable común n (valor inicial 0) entre dos hilos que respectivamente incrementen y decrementen el mismo número de veces a n. Lance ambos hilos concurrentemente utilizando y compruebe que, aunque el valor teórico final debe ser cero, en la práctica no tiene por qué ser así. Guarde su código en `hebra.java` y `Usa_hebra.java`. Escriba (utilizando LaTeX vía *OverLeaf*) una corta tabla de prueba en `tabla.pdf` donde recogerá el número de iteraciones que realizaron los hilos y el valor final obtenido para n junto con sus impresiones acerca de todo ello.

2. Utilizando implementación de la interfaz `Runnable`, cree una condición de concurso sobre una objeto común que albergará una variable entera n con valor inicial 0. La clase que modela al objeto tendrá dos modificadores que respectivamente incrementen y decrementen a la variable n, y un observador para conocer su estado. Lance ambos hilos (uno que incremente y otro que decremente) concurrentemente utilizando y compruebe que, aunque el valor teórico final de n debe ser cero, en la práctica no tiene por qué ser así. Guarde su código en `tareaRunnable.java` y `Usa_tareaRunnable.java`. Escriba (utilizando LaTeX vía *OverLeaf*) una corta tabla de prueba en `tabla.pdf` donde recogerá el número de iteraciones que realizaron los hilos y el valor final obtenido para n

[1] Una condición de concurso sobre un recurso compartido se produce cuando dos o más *threads* acceden al recurso de forma concurrente **sin control de exclusión mutua**.

junto con sus impresiones acerca de todo ello.

3. Se desea realizar el escalado de un vector de números enteros de 10^6 componentes. Escriba un programa secuencial `escalaVector.java` que haga el trabajo. Ahora, escriba una nueva versión paralela multihebrada y llámela `escalaVPar.java`. Escriba un documento que incluya una tabla de análisis `tablaCPU.java` que deberá recoger de forma aproximada los picos de uso máximo de la CPU como una función del tamaño del vector $(10^6, 2 \times 10^6, 3 \times 10^6 ...)$ y del tipo de procesamiento empleado.

4. Escriba en `cuentaCorrienta.java` una clase que modele una cuenta corriente. Incorpore al menos los atributos número de cuenta y saldo, y los métodos depósito y reintegro. Simule ahora, utilizando hilos mediante tareas por implementación de la interfaz `Runnable` a una red de cajeros automáticos, donde cada cajero (guardar en fichero `cajero.java`) realiza una operación de ingreso o de reintegro sobre un cuenta corriente. Provoque ahora una condición de concurso de los hilos contra una instancia de la clase anterior, de forma que la suma neta de las operaciones de todos ellos sea igual a 0. En esta situación, el saldo inicial de la cuenta debería haber permanecido constante. Compruebe que no tiene por qué ser así. Guarde la clase que crea y simula a los cajeros en `redCajeros.java`.

5. Escriba ahora una condición de concurso de dos tareas sobre una variable compartida, utilizando para modelar a la tareas expresiones λ.

2. Procedimiento y Plazo de Entrega

Se ha habilitado una tarea de subida en *Moodle* que le permite subir cada fichero que forma parte de los productos de la práctica de forma individual en el formato original. Para ello, suba el primer fichero de la forma habitual, y luego siga la secuencia de etapas que el propio *Moodle* le irá marcando. Recuerde además que:

- Los documentos escritos que no sean ficheros de código deben generarse **obligatoriamente** utilizando Latex, a través del editor *OverLeaf*, disponible en la nube. Tiene a su disposición en el Campus Virtual un manual que le permitirá desarrollar de forma sencilla y eficiente documentos científicos de alta calidad. Puede encontar el citado manual en la sección dedicada a Latex en el bloque principal del curso virtual. El url de *OverLeaf* es: `https://www.overleaf.com/`

- No debe hacer intentos de subida de borradores, versiones de prueba o esquemas de las soluciones. *Moodle* únicamente le permitirá la subida de los ficheros por **una sola vez**.

- La detección de plagio (copia) en los ficheros de las prácticas, o la subida de ficheros vacíos de contenido o cuyo contenido no responda a lo pedido con una extensión mínima razonable, invalidará plenamente la asignación, sin perjuicio de otras acciones disciplinarias que pudieran corresponder.

- El plazo de entrega de la práctica se encuentra fijado en la tarea de subida del Campus Virtual.
- Entregas fuera de este plazo adicional no serán admitidas, salvo causa de fuerza mayor debidamente justificadas mediante documento escrito.
- Se recuerda que la entrega de todas las asignaciones de prácticas es recomendable, tanto un para un correcto seguimiento de la asignatura, como para la evaluación final de prácticas, donde puede ayudar a superar esta según lo establecido en la ficha de la asignatura.
- Se recuerda que la entrega de todas las asignaciones de prácticas es obligatoria, y requisito indispensable para poder superar las prácticas de la asignatura, y por tanto a la asignatura en sí misma.