Programación Concurrente y de Tiempo Real*
Grado en Ingeniería Informática
Asignación de Prácticas Número 6

Se le plantean a continuación un conjunto de ejercicios sencillos de programación de control de la exclusión mutua y uso de ejecutores, que debe resolver de forma individual como complemento a la sexta 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. En la carpeta de la práctica se le proporciona el código necesario para implantar una solución cliente-servidor simple multihebrada. El inconveniente de esta solución es la necesidad de crear un *thread* para dar servicio a cada petición de un cliente recibida por el servidor. Una solución mucho más elegante es modelar la tarea de servicio mediante objetos `Runnable`, y delegar su ejecuci´on a un *pool* de *threads*. Se pide:

- Reescriba el código del servidor de forma que cada petición de servicio sea atendida por una hebra que procesará un objeto de clase `ThreadPoolExecutor`, y guárdelo en `ServidorHiloconPool.java`.
- Escriba ahora una versión del cliente que genere un número fijo de peticiones al servidor que irán creciendo, y guárdela en `clienteMultiple.java`.

2. Deseamos que varias hebras soportadas mediante herencia de la clase `Thread` escriban datos en un objeto de array de forma segura. Provea una

solución llamada `arrSeguro.java` que cumpla con la especificación descrita, utilizando cerrojos `synchronized`.

3. Escriba una clase `heterogenea.java` que tendrá dos atributos n, m a incrementar mediante **métodos diferentes**. Proteja los métodos que gestionan a n mediante `synchronized`, mientras que los métodos que gestionan a m no tendrán control alguno. A continuación, escriba en `usaHeterogenea.java` un programa donde múltiples hebras accedan a un objeto de clase `heterogenea` concurrentemente, y compruebe, estudiando los valores finales de n y m, que en mismo objeto de Java pueden existir regiones de código bajo exclusión mutua y sin ella.

4. Escriba, utilizando `synchronized`, un código donde tres hebras diferentes (soportadas por herencia de `Thread`) entren en *deadlock*. Guarde el código en `deadlock.java`

5. Deseamos calcular el producto de los números resultantes de sumar las filas de una matriz de enteros de diez filas por cien columnas. Utilizando tareas soportadas por `Callable` y la interfaz `Future`, escriba en `productoSumaFilas.java` un programa en Java que efectúe la tarea. En este caso, prescinda de la ecuación de Subramanian, y utilice diez tareas concurrentes.

6. Rescate ahora todas las condiciones de concurso no controladas que escribió en la asignación número 2, y transfórmelas en versiones concurrentemente seguras utilizando cerrojos `synchronized`. Guarde las nuevas versiones en ficheros con el mismo nombre que ya utilizó para esa asignación, añadiendo a esos nombres la palabra «seguro».

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.