Programación Concurrente y de Tiempo Real
Grado en Ingeniería Informática
Asignación de Prácticas Número 8

En esta asignación aplicará control de exclusión mutua y sincronización, utilizando para ello el API estándar de Java, efecuando la implementación con monitores a soluciones de problemas clásicos de la concurrencia. `Documente todo su código con etiquetas (será sometido a análisis con javadoc)`.

1. Enunciados

1. Una forma elegante de modelar una cuenta corriente compartida entre varios titulares en Java es utilizar un monitor implementado con el API estándar. Cada titular puede depositar o retirar dinero de la cuenta. El saldo actual de la cuenta es la suma de todos los depósitos realizados menos la suma de todos los reintegros realizados hasta la fecha. El balance nunca debe llegar a ser negativo. Los titulares de cuentas se representan como hebras. Una hebra que desea hacer un depósito nunca tiene por qué esperar (excepto por exclusión mutua en el acceso a la cuenta), pero una hebra que desea efectuar un reintegro tiene que esperar hasta que haya fondos suficientes. Desarrolle un monitor para resolver este problema. El monitor debe tener dos procedimientos: `ingreso (cuantía)` y reintegro(cuantía). Suponga que el argumento de ambos procedimientos es siempre positivo. Guarde el monito en `cuentaCompartida.java`, y escriba un diseño de hebras depositantes y reintegrantes que haga uso del mismo en `cuentaCompartida.java`.

2. Suponga que n tareas concurrentes comparten tres impresoras. Antes de poder utilizar una impresora, la tarea que desea imprimir debe solicitar una impresora disponible mediante un método `int pedirImpresora()`, que devuelve el número de la impresora asignada. Tras utilizar la impresora, esta se libera mediante un método `void liberarImpresora(int n)`, cuyo parámetro es la impresora que se libera. Desarrolle un monitor en `monitorImpresoras.java`, y un diseño de hebras que lo utilice en `usamonitorImpresoras.java`