Métodos sincronizados/Locks

1.- Varios procesos (N) compiten por utilizar unos cuantos recursos Rec en exclusión mutua. El uso correcto de los recursos es gestionado por una clase **Control**. Cada proceso pide un numero de recursos llamando al método **qrecursos(id,num),** donde **id** es el identificador del proceso que hace la petición y **num** es un número entero que especifica el número de recursos que pide el proceso. Después de usar los recursos, el proceso los libera ejecutando el método **librecursos(id,num).** Para asignar los recursos **Control** utiliza la técnica FCFS (First Come, First Serve) que significa que los procesos son servidos siempre en el orden en el que han realizado sus peticiones. Así un proceso que pide **num** recursos debe esperarse si hay otros procesos esperando aún cuando existan **num** recursos disponibles en el sistema. Implementa la clase **Control**, y varias hebras usuarios que realicen las peticiones a **Control.**

2.- Supongamos que N niños asisten a una fiesta de cumpleaños, en la que se les ofrece de forma repetida raciones de tarta de chocolate que se encuentran en una bandeja. Como las raciones se acaban con frecuencia, existe un pastelero que, cuando lo avisan, pone una nueva tarta en la bandeja (suponemos que cada tarta da lugar a un número fijo R de raciones). Cuando un niño quiere una ración de tarta, va a la bandeja y la coge, si no está vacía. Si ve que la bandeja está vacía, avisa al pastelero para que traiga una nueva tarta. Implementa este sistema utilizando métodos sincronizados o locks, teniendo en cuenta que deben satisfacerse las dos condiciones de sincronización siguientes:

-CS1: Un niño que quiere una ración de tarta debe esperar si la bandeja está vacía.

-CS2: El niño que ha avisado al pastelero debe coger la ración de tarta antes que cualquier otro de la fiesta.

-CS3: El pastelero no pone una nueva tarta en la bandeja, hasta que la bandeja está vacía.

Supón que el pastelero es perezoso, por lo que mientras que no lo avisan se queda durmiendo un ratito. El esqueleto del sistema se encuentra en el campus virtual. Cuando un niño quiere una ración llama al método public void quieroRacion(int id) del objeto bandeja.

Por otro lado, el pastelero llama al método public void tarta() para poner una nueva tarta sobre la bandeja. Supón que la bandeja está inicialmente vacía.

3.- Se dispone de una sala donde distintas parejas tienen citas, para ello un hombre y una mujer tienen que encontrarse (sincronizarse) en la sala. Toda la sincronización del sistema se realiza en un objeto Parejas. Cuando el hombre id quiere tener una cita llama al método *llegaHombre(int id),* y lo mismo para la mujer id que llama a *llegaMujer(int id)* cuando quiere tener una cita.

Las condiciones de sincronización del sistema son:

- Un hombre no puede entrar en la sala, si ya hay otro dentro
- Un hombre tiene que esperara en la sala, si no hay ya dentro una mujer
- Las condiciones de sincronización para las mujeres son las mismas