UNIVERSIDAD DE MÁLAGA
Dpto. Lenguajes y Ciencias de la Computación

Programación de Sistemas y Concurrencia
Control 23/5/2013

APELLIDOS _____ NOMBRE _____

DNI _____ ORDENADOR _____ GRUPO _____

1.- SEMÁFOROS.

a) Una empresa debe abordar la reparación de unas máquinas de una sala de un edificio contaminado con sustancias radiactivas. El número de trabajadores de la empresa es de **10** pero en dicha sala únicamente pueden estar **tres** trabajadores a la vez y durante un tiempo limitado (TMAX) para evitar la contaminación. Transcurrido ese tiempo saldrán de la habitación y, pasado un tiempo (TDES), volverán a intentar entrar en la habitación para volver a trabajar. Para poder trabajar en esa habitación cada trabajador necesita adquirir antes de intentar entrar en la sala **2 herramientas**. La empresa tiene **5 herramientas** en total. Cada vez que el trabajador tiene que salir a descansar libera sus herramientas para que otros trabajadores puedan tomarlas y entrar en la sala. La empresa necesita llevar un control de entradas a la sala por lo que en todo momento debe saber cuántas veces se ha trabajado en la sala. Finalmente, existe una hebra adicional de control de costes que cada vez que entran 10 trabajadores muestra por pantalla un mensaje de aviso e incrementa el coste del trabajo en 1000 euros. **Implementar una solución a este problema usando sólo SEMÁFOROS BINARIOS**.

Con el fin de facilitar la implementación del sistema, en el campus virtual se encuentra un fichero Planta.java con una solución *parcialmente* implementada. La clase Planta tiene dos clases estáticas implementadas. La clase Trabajador tiene el comportamiento de los trabajadores descrito arriba. En concreto, su método run() consta de cuatro operaciones que se repiten de forma ininterrumpida:

- *entraSala(int id);* llamado por el trabajador id para entrar en la sala contaminada.
- Thread.*sleep(TiempoEnSala);* tiempo que pasa el operador dentro de la sala
- *saleSala(id);* llamado por el trabajador id antes de salir de la sala contaminada
- Thread.*sleep*(*TiempoDescanso*); tiempo que pasa el operador fuera de la sala.

Por otro lado, Planta también tiene anidada a la clase estática Contador cuya tarea es la de imprimir en la pantalla el incremento del coste del trabajo cada vez que entran 10 trabajadores en la sala.

Observa que en la clase Planta puede haber métodos con la cabecera incompleta.

Cada estudiante puede decidir si quiere seguir la estructura sugerida en esta solución, o implementar la solución de otra manera.

2.- MONITORES

a) .- El mostrador de queso de un supermercado está rodeado continuamente por clientes hambrientos. Hay dos tipos de clientes: los **clientes descarados** que se cuelan y piden que se les atienda; y los **clientes amables** que esperan pacientemente su turno. Suponiendo que siempre queda queso en el mostrador, modela un sistema en el que hay un número fijo de clientes descarados y amables. Cada cliente es una hebra que pide queso y cuando es atendido se marcha del supermercado. Además existe un proceso dependiente que atiende la petición de los clientes, según el tipo al que pertenezcan. Es decir, de todos los clientes que quieren queso, el dependiente atiende primero a los descarados y luego a los amables. Cuando ha atendido a todos los clientes, el dependiente se queda dormido. Implementa este sistema utilizando **métodos sincronizados o locks/condiciones**.

Con el fin de facilitar la implementación del sistema, en el campus virtual se encuentra un fichero Tienda.java con la solución parcialmente implementada. La clase Tienda tiene tres clases estáticas anidadas. Las clases ClienteAmable, ClienteDescarado representan los dos tipos de cliente. Su método run consta de:

-Thread.*sleep*(*r*.nextInt(2000)); que representa el tiempo aleatorio que pasan los clientes antes de entrar en la tienda

- *qQuesoDescarado*(id);/ *qQuesoAmable*(id); llamado por los respectivos clientes (con identificador id) cuando entran en la tienda para pedir su ración de queso. Una vez terminado el método el cliente debe haber conseguido su ración.

Finalmente, la clase Dependiente representa el comportamiento del dependiente de la tienda. En su método run() ejecuta de forma ininterrumpida el método siguiente(), en el que atiende a uno de los clientes si hay alguno en la tienda, o se duerme, en otro caso.

Observa que en la clase Tienda puede haber métodos con la cabecera incompleta

Cada estudiante puede decidir si quiere seguir la estructura sugerida en esta solución, o implementar la solución de otra manera

Notas para hacer el control:

1.- Crea un proyecto **`prControl`**
2.- Crea dos paquetes en **`prControl`** denominados **`planta`** y **`tienda`**, y deja los fuentes de cada uno de los ejercicios en su paquete correspondiente.
3.- **Pon tu nombre en todos tus fuentes**.
4.- Cuando termines, haz un fichero comprimido (.zip, .rar, .jar) **con los fuentes** de tus ejercicios y súbelos a la tarea del campus virtual creada para este fin.
5.- En el control, no pueden utilizarse ni apuntes, ni ejercicios de clase. Sólo está permitido consultar la documentación de java.