Programación Concurrente

Práctica 6: Monitores

Ejercicio 1. Implementar un monitor `Contador` que permita incrementar y decrementar concurrentemente de forma segura.

Ejercicio 2. Definir un monitor Semáforo que implemente las operaciones de un semáforo (`acquire` y `release`). ¿Garantiza su solución ausencia de starvation?

Ejercicio 3. Implementar lo siguiente utilizando monitores:

a) Un buffer de números enteros cuya dimensión está prefijada al momento de creación del mismo.

b) Una clase productor que agregue números naturales consecutivos a un buffer dado al momento de creación.

c) Una clase consumidor que muestre por pantalla los valores que toma de un buffer pasado al momento de creación del mismo.

d) Un programa que cree un buffer de dimensión 2 y active concurrentemente un consumidor y un productor.

Ejercicio 4. Se desea implementar usando monitores un `secuenciador ternario` para coordinar a N threads. Un secuenciador ternario provee tres operaciones `primero`, `segundo`, `tercero`. La idea es que cada uno de los threads puede invocar a cualquiera de las operaciones. El secuenciador alternará cíclicamente la ejecución de `primero`, luego de `segundo`, y finalmente `tercero`.

Ejercicio 5. Se desea implementar usando monitores una barrera para coordinar a N threads. Una barrera provee una única operación denominada `esperar`. La idea es que cada uno de los N threads invocarán la operación `esperar` una vez y el efecto de invocarla es que el thread se bloquea y no puede continuar hasta tanto los restantes threads invoquen a la operación esperar. Por ejemplo, si miBarrera es una barrera para coordinar 3 threads, el uso de miBarrera en los siguientes threads

```
thread T1: { print('a'); miBarrera.esperar(); print(1); }
thread T2: { print('b'); miBarrera.esperar(); print(2); }
thread T3: { print('c'); miBarrera.esperar(); print(3); }
```

garantiza que todas las letras se mostrarán antes de los números. Dar una implementación para el monitor barrera tal que una vez alcanzado el cupo la barrera quede levantada, es decir, que no vuelva a bloquear invocaciones al método esperar.

Ejercicio 6. En una red de energía inteligente los usuarios conectados pueden comportarse como consumidores o productores de energía (no ambos al mismo tiempo). Para registrar los cambios se desea utilizar un sistema implementado con un monitor. Por ejemplo el siguiente fragmento de código muestra un usuario que se comporta como consumidor por una hora y como productor por 2 horas:

```
grid.startConsuming();
sleep(1h);
grid.stopConsuming();
grid.startProducing();
sleep(2h);
grid.stopProducing();
```

Resuelva los siguientes escenarios usando monitores:

a) Los usuarios siempre pueden comportarse como productores, pero sólo pueden comportarse como consumidores si hay una cantidad igual o mayor de productores. Además se debe garantizar que un productor no puede dejar de producir si no hay capacidad ociosa en la red, es decir, no puede dejar de producir si al hacerlo algún consumidor quedaría sin suministro.

b) Modifique su solución de forma tal que los usuarios no puedan comportarse como productores si la red tiene más de N productores ociosos, es decir, que el cambio a productor genere una espera mientras haya al menos N productores más que consumidores.

c) Extienda la solución anterior de forma tal que se priorice la salida de productores por sobre la entrada de consumidores nuevos.

Ejercicio 7. En la pizzería del barrio se fabrican dos tipos de pizzas: las chicas y las grandes. El maestro pizzero va colocando de a una las pizzas listas en una barra para que los clientes se sirvan y pasen luego por la caja. Lamentablemente todos los clientes tienen buen apetito y se comportan de la siguiente manera: prefieren siempre tomar un pizza grande, pero si no lo logran se conforman con dos pequeñas. Los clientes son mal educados y no esperan en una cola (todos compiten por las pizzas). Dar una solución en donde cada cliente es modelado como un thread. Como mecanismo de sincronización, utilizar monitores.

Ejercicio 8. El comité organizador de una conferencia posee una sala para exposición de charlas. Las personas que desean asistir a una charla entran en la sala y esperan hasta que la charla comienza. Las charlas empiezan cuando el orador llega a la sala. Por respeto los asistentes no se retiran hasta que la charla termina ni entran cuando la charla ya ha comenzado. La sala posee una capacidad de 50 personas, sin contar al orador.

De una solución usando monitores que modele los siguientes escenarios:

a) Se dispone de una sala para dar repetidas veces la misma charla. El orador descansa 5 minutos entre charla y charla. Si al momento de iniciar la charla el auditorio está vacío el orador descansa otros 5 minutos esperando que lleguen oyentes. Cuando el auditorio se llena los asistentes deben esperar a que comience la siguiente charla luego del descanso del orador.

b) Al igual que en el punto a) se dispone de una sala para dar tres charlas distintas, pero en este caso los oradores esperan a que haya al menos 40 personas en el auditorio.