CC4302 Sistemas Operativos – Tarea 1 – Semestre Otoño 2021 – Prof.: Luis Mateu

En un pub del barrio universitario los y las estudiantes se reunen en la noche a tomar cerveza. Cada cierto tiempo, los/las estudiantes deben visitar el baño. El baño es amplio y admite un número ilimitado de estudiantes. Como es lógico existe una restricción: damas y varones no deben compartir el baño.

Las damas y varones se representan mediante threads de nSystem. Para coordinar el acceso al baño las damas invocan *entrar(DAMA)* antes de entrar al baño y llaman a *salir(DAMA)* después de salir. Lo mismo hacen los varones con las funciones *entrar(VARON)* y *salir(VARON)*. El problema consiste en programar las funciones *entrar*, *salir* e *ini_pub* de manera que se cumpla la exclusión mutua de varones y damas dentro del baño. Este es un problema similar al problema de los lectores/escritores.

Se debe implementar una política de entradas alternadas entre damas y varones. Esto significa que si las damas están ocupando el baño, cuando sale la última dama, entran todos los varones en espera. Simétricamente si los varones están ocupando el baño, cuando sale el último varón, entran todas las damas en espera. Para evitar la hambruna se requiere que cuando los varones están ocupando el baño pero hay damas en espera, ningún otro varón podrá entrar. Del mismo modo cuando las damas están ocupando el baño pero hay varones en espera, ninguna otra dama podrá entrar.

Por ejemplo, en un instante Ana y María están ocupando el baño y no hay varones en espera. Entonces llega Rosa y sí entra de inmediato. Luego llega Pedro que debe esperar porque hay damas ocupando el baño. Más tarde llega Silvia que también debe esperar porque hay un varón en espera. Entonces llega Diego después de Silvia y también debe esperar. Cuando Ana, María y Rosa desocupan el baño, entran Pedro y Diego, a pesar de que Diego llegó después de Silvia. Silvia continúa esperando hasta que Pedro y Diego salgan. Si llega Juan debe esperar porque está Silvia esperando el baño.

Parte a.- (2 puntos) En los archivos adjuntos, el archivo *pub.c* contiene una implementación incorrecta del problema. Cuando se ejecuta con pSystem, ***a veces*** se entrega un mensaje de error indicando que no se cumple la exclusión mutua de varones y damas. Nunca he logrado que se reporte el mismo error con nSystem. Demuestre que esta solución es incorrecta confeccionando un diagrama de threads para un caso de ejecución en donde efectivamente no se cumple la exclusión mutua de varones y damas. Debe considerar que en nSystem y pSystem los tickets de los semáforos siempre se otorgan por orden de llegada.

Parte b.- (4 puntos) Reprograme la solución dada en *pub.c* de modo que funcione correctamente siempre. Ud. debe usar la siguiente metodología:

- Debe emplear solamente los semáforos de nSystem como herramienta de sincronización. No puede usar otras herramientas de sincronización como por ejemplo monitores.
- Use 2 colas globales *fifoqueues*: una para colocar en espera a las damas y la otra para colocar en espera a los varones.
- Cuando una dama debe esperar para ingresar al baño, cree un semáforo con 0 tickets y encólelo en la *fifoqueue* para las damas en espera. Para suspender el thread solicite un ticket a ese semáforo. Haga lo mismo con los varones que deben esperar pero usando la *fifoqueue* para varones. Los encabezados de las operaciones de las *fifoqueues* están en *$NSYSTEM/include/fifoqueues.h*.
- Cuando sale el último varón y hay damas en espera en la *fifoqueue* para damas, extraiga todos los semáforos en esa cola y deposite un ticket en cada uno de ellos para hacer que las damas ingresen al baño. Lo mismo cuando sale la última dama, pero vaciando la cola de varones.
- En la función *entrar*, no olvide destruir el semáforo creado para evitar memory-leaks.
- No olvide usar un semáforo para garantizar la exclusión mutua en el acceso a las variables globales.
- Pruebe su tarea con los comandos *make debug* y *make optimize* con nSystem y pSystem y *make run-valgrind-ddd* y *make run-drd-ddd* solo con pSystem. Lea las instrucciones en el archivo *Makefile*. Su tarea será rechazada si no pasa exitosamente cualquiera de estos comandos. *Valgrind* no debe reportar memory-leaks y *drd* no debe reportar dataraces.
- No se confíe porque *drd* no reporta dataraces. En general no los reporta cuando se usan semáforos para la sincronización.

Archivos

Descargue de U-cursos el archivo *t1.zip* y descomprímalo. Ahí encontrará:

- Un *Makefile* para compilar su tarea. Ud. debe definir la variable de ambiente NSYSTEM con la ubicación de *nSystem* o *pSystem* en su computador.
- *pub.h*: archivo de encabezados.
- *test-pub.c*: test de prueba.
- *pub.c* y *pub-incorrecta.c*: la solución incorrecta del problema. Programe en *pub.c* la solución correcta.

Entrega

Ud. debe entregar mediante U-cursos el archivo *pub.c*. Se descontará medio punto por día hábil de atraso.