Fundamentos de los Sistemas Operativos
Ficha de entrega de práctica

**: campo obligatorio*
***IMPORTANTE:** esta ficha no debe superar las DOS PÁGINAS de extensión*

Grupo de prácticas*: 43 (Viernes 8:30 – 10:30, Alexis Quesada)
Miembro 1: Aitor Ventura Delgado
Miembro 2:

Número de la práctica*: 4 | **Fecha de entrega*:** 06.05.2020

Descripción del trabajo realizado*
Primera parte: modifiqué el archivo test_hilos.c, añadí condiciones y mutex en los métodos de productor y consumidor. Todo lo añadido se encuentra bajo el comentario //Añadido por el alumno. Para que existiera una ejecución "bonita" del programa y no encontrásemos ningún bucle infinito, hice que aparecieran mensajes de error. Sin embargo, esto no se trata de ningún error, pero lo hice para tener una ejecución que finalizara dado cualquier tipo de parámetros.

Se ha añadido un fichero .txt que indica las diferencias entre un ejecutable u otro.

Segunda parte: elegí el problema de los caníbales y misioneros. Consta de ocho métodos, sin incluir el main. Al ejecutarse, hay que introducir por la línea de órdenes tanto el número de misioneros como de caníbales. Además, se modificó la estructura del buffer para tenerlo como bote en este caso. (Teniendo métodos de insertar item ahora como subirBote, extraer item como bajarBote, etc…)

El funcionamiento principal se encuentra en las rutinas llegaMisionero() y llegaCanibal(). Estos se tratan de una "biografía" tanto de misioneros como de caníbales, respectivamente. Se suben al bote, viajan, y se bajan, siempre evadiendo que se suban 2 misioneros y 1 caníbal para evitar la muerte de este último, así como mostrando por pantalla el evento en el que se encuentran. A través de variables condición y cerrojo se hace la sincronización entre hilos, evitando situaciones mencionadas en el código en comentarios. He intentado documentar el código con tanto comentario posible para evitar cualquier posible confusión del código.

Tenemos dos métodos que checkean errores, para tener el manejo de errores organizado. Uno se trata de las creaciones de hilos y de los join, que requieren de un parámetro extra, y el otro no requiere el parámetro extra sino únicamente el código de error que se le pasa.

Se ha separado en módulos, teniendo así por un lado la estructura del bote, por otro el funcionamiento de los caníbales y misioneros, y por otro el archivo main que inicia la ejecución.

De la misma manera, se ha añadido un .txt que indica las diferencias entre un ejecutable u otro.

Horas de trabajo invertidas* Miembro 1: 17+ **Miembro2:**

Cómo probar el trabajo*
Se entrega un script en ambos casos, para poder tener varias pruebas en un mismo lugar. Para tener permiso de ejecución de estos, usamos: chmod +x *script* ó bien chmod 700 *script*.

Primera parte:
gcc -o practica4 test_hilos.c buffer_circulos.c -lpthread
./practica4 capacidad productores consumidores
./test_buffer.sh &> res_testbuffer.txt
Y abrimos res_testbuffer.txt para ver los resultados.

Grupo de prácticas*: 43 (Viernes 8:30 – 10:30, Alexis Quesada)
Miembro 1: Aitor Ventura Delgado
Miembro 2:

Número de la práctica*: 4	**Fecha de entrega*:** 06.05.2020

Segunda parte:
gcc -o misioneros misioneros.c bote.c main.c -lpthread
./misioneros misioneros caníbales
./test_misioneros.sh &> res_testmisioneros.txt
Y abrimos res_testmisioneros.txt para ver los resultados.

Si se quisiera compilar el código de errores, compilar de la siguiente manera
gcc -o misioneroserror misioneroserror.c bote.c main.c -lpthread
./misioneroserror misioneros caníbales

Incidencias
(errores no resueltos, anomalías, cualquier cosa que se salga de lo normal)

Comentarios
Si se desea que la ejecución de caníbales y misioneros no termine “de manera bonita”, séase que siempre se termina, entonces remover la línea indicada en los comentarios del código.