Práctica 5: Sincronización de hilos de un mismo proceso

Índice

1 Objetivos

El objetivo de esta práctica es afianzar nuestro conocimiento de los mecanismos de sincronización de hilos que ofrece un sistema POSIX y sus esquemas de uso. En la práctica haremos uso de: mutex y variables de condición para sincronizar hilos de un mismo proceso.

El archivo ficheros_p5.tar.gz contiene una serie de ficheros que pueden ser usados como punto de partida para el desarrollo los ejercicios de esta práctica, así como unos makefiles que pueden ser usados para la compilación de los distintos proyectos.

2 Desarrollo de la práctica

Se desea diseñar el sistema de control del aforo de una discoteca en la que el número máximo de clientes que pueden estar dentro en un momento dado (aforo) es N. Además, hay dos tipos de clientes: los vip y los normales. Las reglas que debe cumplir el sistema son las siguientes:

- Si el aforo está completo los nuevos clientes deberán eseperar a que salga algún cliente de la discoteca para poder entrar.
- Si hay esperando clientes vip y clientes normales, se les dará prioridad a los clientes vip. Los clientes normales deberán por tanto esperar a que entren los vip primero.
- Los clientes entrarán de uno en uno en orden estricto de llegada según su grupo (normales o vips), mientras el número de ocupantes de la discoteca sea menor que el aforo (N).

Se debe simular el sistema con un programa que cree M hilos que representan a los clientes de la discoteca. Estos hilos deberán ejecutar una función de entrada llamada `client`, con la siguiente estructura:

```
void *client(void *arg)
{
    ...

    if ( isvip )
        enter_vip_client( ... );
    else
        enter_normal_client( ... );
    dance( ... );
    exit_client( ... );
```

```
    ...
}
```

En su creación a cada hilo se le pasarán dos argumentos enteros:

- id: un identificador de la tarea que corresponde con el orden de creación del hilo.
- isvip: un valor que indique si el cliente es vip o no.

Diseñar las funciones **enter_vip_client**, **enter_normal_client** y **exit_client** de forma que garanticen las condiciones de funcionamiento del sistema descritas arriba. Añadir en estas funciones mensajes por consola con *printf* que permitan hacer el seguimiento del programa, indicando en cada mensaje el id del cliente y lo que está haciendo.

El programa principal recibirá por la línea de comandos el nombre de un fichero que contendrá el número de clientes a crear (M) y el carácter vip o normal de cada uno de estos clientes. El formato esperado de este fichero es el siguiente:

- Un fichero ascii organizado por líneas
- La primera línea indica el número de clientes a crear (M)
- Las M lineas siguientes (una por cliente) tomarán el valor 1 o el valor 0 para indicar el carácter vip de dicho cliente (nótese que el 1 y el 0 son caracteres ascii).

Por ejemplo, para 5 clientes con 3 no vip y 2 vip el contenido del fichero podría ser:

```
5
0
0
1
0
1
```

El fichero puede parsearse fácilmente utilizando la función de la biblioteca estándar de C *fscanf*, consultar su página de manual.