SISTEMAS OPERATIVOS - LABORATORIO
14 de junio de 2024

Nombre _____ DNI _____

Apellidos _____ Grupo _____

INSTRUCCIONES

- Se debe entregar una carpeta individual con el código de cada apartado (1A, 1B, etc.), para así garantizar la evaluación independiente de cada uno.
- Si el código no compila o su ejecución produce un error grave la puntuación de ese apartado será 0.

Cuestión 1. (5 puntos) Desarrollar un programa, `library.c`, que simule un sistema de control de acceso a una biblioteca. La biblioteca tiene un aforo máximo de `N` personas. Hay dos tipos de usuarios: estudiantes y profesores. Las reglas que debe cumplir el sistema son las siguientes:

1. Si el aforo está completo, los nuevos usuarios deberán esperar a que salga alguien de la biblioteca para poder entrar.
2. Si hay esperando profesores y estudiantes, se les dará prioridad a los profesores. Los estudiantes deberán esperar a que entren los profesores primero.
3. Los usuarios entrarán de uno en uno en orden estricto de llegada según su grupo (estudiantes o profesores), mientras el número de ocupantes de la biblioteca sea menor que el aforo (`N`).
4. La biblioteca puede cerrar por un periodo vacacional durante el cual no puede entrar nadie (ni profesores ni estudiantes). Los usuarios que intenten entrar durante este periodo deberán esperar hasta que la biblioteca reabra. Los usuarios que intenten salir podrán hacerlo con normalidad.

Apartado A (2,5 puntos) Adaptar la práctica 5 de forma que se garantice las condiciones de funcionamiento del sistema descritas en las reglas 1 a 3. Por simplicidad, se supondrá `N=3`, que la biblioteca permanecerá abierta todo el tiempo, y que cada usuario permanecerá dentro de la biblioteca 2 segundos. Se usará el siguiente fichero de entrada, `input.txt` , con 7 estudiantes y 3 profesores:

```
Unset
10
0
0
0
0
0
1
1
1
0
0
```

Apartado B (2,5 puntos) Añadir al ejercicio del apartado anterior el funcionamiento correspondiente a la regla número 4 indicada más arriba. Para ello el programa principal creará un hilo adicional cuya misión es cerrar la biblioteca tras 4 segundos abierta y volver a abrirla tras 2 segundos cerrada, repitiendo este patrón indefinidamente. El programa arrancará con la biblioteca abierta y forzará la terminación de este hilo con `pthread_cancel(pthread_t)` una vez que todos los usuarios hayan salido de la biblioteca.
Un ejemplo de simulación, siguiendo las especificaciones anteriores, podría ser:

```
Unset
$ make
$ make test-library
$ ./library input.txt
```

```
User 0 (student) waiting on the queue.
User 0 (student) is reading books for 2 seconds.
User 2 (student) waiting on the queue.
User 4 (student) waiting on the queue.
User 2 (student) is reading books for 2 seconds.
User 1 (student) waiting on the queue.
User 7 (professor) waiting on the queue.
User 7 (professor) is reading books for 2 seconds.
User 8 (student) waiting on the queue.
User 5 (professor) waiting on the queue.
User 6 (professor) waiting on the queue.
User 9 (student) waiting on the queue.
User 3 (student) waiting on the queue.
User 0 (student) leaves the library.
User 7 (professor) leaves the library.
User 2 (student) leaves the library.
User 5 (professor) is reading books for 2 seconds.
User 6 (professor) is reading books for 2 seconds.
User 4 (student) is reading books for 2 seconds.
Library is closing for vacation.
User 5 (professor) leaves the library.
User 6 (professor) leaves the library.
User 4 (student) leaves the library.
Library is now open after vacation.
User 1 (student) is reading books for 2 seconds.
User 8 (student) is reading books for 2 seconds.
User 9 (student) is reading books for 2 seconds.
User 9 (student) leaves the library.
User 8 (student) leaves the library.
User 1 (student) leaves the library.
User 3 (student) is reading books for 2 seconds.
Library is closing for vacation.
User 3 (student) leaves the library.
```

Cuestión 2. (5 puntos) Se desea crear un programa que permita clasificar automáticamente una serie de imágenes en dos directorios diferentes, el directorio para imágenes cuyo tamaño sea superior a un umbral, `Folder_01`, y el directorio para imágenes cuyo tamaño sea inferior a dicho umbral, `Folder_02`. El umbral por defecto será de `300KB`.

Apartado A (3 puntos) Implementar un programa que recorra todos los ficheros del directorio actual realizando la búsqueda de aquellos que tengan la extensión `.png`. Por cada imagen se generará un proceso que cree un enlace rígido (*hard link*), mediante la llamada al sistema `link`, entre el fichero que se encuentra en el directorio original y la carpeta `Folder_01` o `Folder_02`, según el criterio de tamaño especificado. Las carpetas se deben crear manualmente antes de ejecutar el programa. Tanto las rutas a los directorios como el umbral serán variables fijas en el código, no argumentos.
Para verificar el funcionamiento se utilizarán las imágenes disponibles en el Campus Virtual. La salida que se obtendrá tras ejecutar el comando `ls -l` en el directorio en el que se encuentra el programa y se almacenan las imágenes será similar a la siguiente:

```
Unset
drwxrwxr-x  2 user user   4096 jun 14 10:10 Folder_01
drwxrwxr-x  2 user user   4096 jun 14 10:10 Folder_02
-rw-rw-r-- 10 user user 230037 jun 14 10:33 Img1.png
-rw-rw-r-- 10 user user 369237 jun 14 10:33 Img2.png
-rw-rw-r-- 10 user user 333554 jun 14 10:34 Img3.png
-rw-rw-r--  5 user user    273 jun 14  2023 Makefile
-rwxrwxr-x  1 user user  17808 jun 14 10:36 space
-rw-rw-r--  6 user user   2951 jun 14 10:36 space.c
```

Y por tanto, la salida del mismo comando para los directorios `Folder_01` y `Folder_02` , tras ejecutar el programa, será similar a esta:

`Folder_01`:

```
Unset
-rw-rw-r-- 10 user user 369237 jun 14 10:33 Img2.png
-rw-rw-r-- 10 user user 333554 jun 14 10:34 Img3.png
```

`Folder_02`:

```
Unset
-rw-rw-r-- 10 user user 230037 jun 14 10:33 Img1.png
```

Para implementar este programa, se puede recurrir al código `espacio.c` (`space.c`) de la práctica 3, ejercicio 4. Ejecutando el comando `./espacio *` (`./space *`), tras su modificación, la salida generada debería ser similar a la siguiente:

```
Unset
$ ./space *
4K    Folder_01
4K    Folder_02
228K    Img1.png
Link to ./Img1.png  created successfully
364K    Img2.png
Link to ./Img2.png  created successfully
328K    Img3.png
Link to ./Img3.png  created successfully
4K    Makefile
20K    space
4K    space.c
```

Apartado B (2 puntos) Modificar el programa anterior para que tanto el umbral como las rutas a las carpetas `Folder_01` y `Folder_02` sean argumentos de la función. Para ello recurra a `getopt` (puede utilizar al código del ejercicio 3 de la práctica 2). La llamada a la función deberá presentar un aspecto similar al siguiente:

```
Unset
$ ./space -l './Folder_01/' -s './Folder_02/' -t 200 *
```

Donde el argumento `-l` es el path a `Folder_01`, para imágenes con tamaño superior al umbral; `-s` es el path a `Folder_02`, para imágenes de tamaño inferior al umbral y `-t` es el valor del umbral.