Laboratorio 6: Temporizadores, Tareas Periódicas y Sincronización Simple

Objetivos

- Aprender a configurar temporizadores (*timers*) en espacio de usuario.
- Aprender a configurar hilos como tareas periódicas y a establecer sus prioridades.
- Observar ventajas y desventajas de sincronizar tareas usando únicamente períodos y tiempos de inicio.

Duración: 1½ sesiones

Antes de comenzar los experimentos descritos abajo, presten atención a la "IE3059 - Laboratorio 6, presentación". Estudien los archivos Lab6_files_y_strings.c y Lab6_Timer_functions.c (encuentran los archivos en Canvas). Las funciones mostradas les servirán para completar este laboratorio (y para futuros laboratorios).

Procedimiento

- Cierto texto fue dividido en dos partes. Las líneas impares del texto se guardaron en el archivo llamado **Lab6_primero.txt**, y las líneas pares se guardaron en el archivo **Lab6_segundo.txt**. Deberán crear un programa que reconstruya el texto original, lo despliegue en la terminal, y lo guarde en un tercer archivo, **Lab6_reconstruido.txt**. Para lograr el objetivo, el programa creará tres hilos (*pthreads*) adicionales al hilo principal (*main*), y utilizará un *buffer* común a los hilos. El *buffer* será una cadena de caracteres (un *string*). Puede ser una variable global, aunque se prefiere que sea declarada en el *main* y luego se pase como argumento a los hilos.

- Dos de los *pthreads* adicionales se encargarán de leer los archivos, una línea a la vez (el primer *pthread* abrirá y leerá el archivo **Lab6_primero.txt** y el segundo *pthread* abrirá y leerá el archivo **Lab6_segunto.txt**). Luego de leerse una línea (cadena de caracteres), ésta deberá ser guardada en el *buffer* común. El tercer *pthread* adicional se encargará de leer las cadenas que se guarden en el *buffer*, y las guardará a su vez en un *string array* (arreglo o colección de cadenas). El arreglo debe ser capaz de almacenar hasta 60 cadenas. Por conveniencia, el arreglo sí será una variable global. Finalmente, el hilo principal (*main*) deberá desplegar el texto reconstruido en la terminal, y deberá guardarlo en el archivo **Lab6_reconstruido.txt**.
  Ayuda 1: en sus pruebas, pueden escribir a la terminal lo que vayan leyendo de los archivos en los hilos adicionales 1 y 2, para verificar que la lectura se vaya haciendo correctamente. Sin embargo, en la versión final del programa, sólo el hilo principal debe escribir a la terminal.
  Ayuda 2: el instructor les mostrará la estructura recomendada del *main*.

- Todos los *pthreads* adicionales al *main* deben inicializarse como periódicos (usando un *timer* cada uno), con la misma prioridad, y agendarse de forma alternante, de modo que los datos puedan ser leídos de los archivos y almacenados en el *string array* en la secuencia correcta (**sugerencia**: en los hilos que deben leer los archivos, abran primero esos archivos, y después inicialicen los *timers*). Al terminar de leerse todas las líneas de los archivos, los *pthreads* deben concluir su ejecución. Es decir, las tareas no durarán "para siempre", sino durarán únicamente el tiempo necesario para leer los datos de los archivos y guardarlos en el *string array*.
  Nota: Pueden usar el hecho de que los archivos no tienen más de 30 líneas cada uno.

  Ayuda 3: Consideren el siguiente seudocódigo para los *pthreads* adicionales:

```
Declaración de variables
Recuperación de información pasada al hilo
Asignar prioridad y política de escalonamiento
Abrir archivo (sólo los hilos 1 y 2)
Configurar y arrancar el timer
Esperar a que se cumpla el primer período (que expire el timer)
Ciclo (hasta terminar de leer el archivo / llenar el arreglo)
     Hacer lo que haya que hacer
     Esperar a que expire el timer
Salir del hilo (pthread_exit)
```

- Deberán sincronizar las tareas ajustando sus períodos y sus tiempos de inicio. Prueben distintas combinaciones. Empiecen con períodos relativamente largos (ej. 100 o 200 ms), para asegurarse de que las tareas tengan tiempo de leer y guardar los datos en el *buffer*. Luego, prueben reducir los períodos. **Encuentren los períodos más pequeños que puedan, tales que la reconstrucción del archivo original sea exitosa. Reporten los distintos tiempos probados, incluyendo los menores que les hayan funcionado. Generen una captura de pantalla mostrando el texto reconstruido en una terminal.**

- Recuerden incluir los encabezados (.h) apropiados en sus programas (revisen los archivos .c proporcionados). También recuerden incluir lo necesario a la hora de compilar sus programas (*pthreads)*.

Evaluación:

Deberán subir a Canvas un **reporte (.pdf), el cual deberá incluir los tiempos que usaron, así como la captura de pantalla. Incluyan también una breve discusión de lo realizado, y mencionen ventajas y desventajas del método de sincronización utilizado en esta práctica**. Asegúrense de que el documento esté bien identificado (nombre, carné, curso, número de laboratorio, fecha).
También **deberán subir su código final a Canvas (archivo .c)**. Recuerden incluir sus datos como comentarios al inicio de su código.

Asistencia y trabajo en el lab:	40%
Reporte (.pdf):	30%
Programa (.c):	30%

Fecha límite de entrega: revisar la tarea “Lab 6” en Canvas.

Material de Apoyo

- https://linux.die.net/man/2/timerfd_create
- https://linux.die.net/man/2/sched_setscheduler
- https://linux.die.net/man/2/read