SISTEMAS OPERATIVOS
TRABAJO PRÁCTICO 3:
Comunicación entre Procesos

Objetivos del práctico

Al terminar este trabajo Ud. habrá aprendido a:

- Utilizar las distintas llamadas al sistema que nos permiten implementar la comunicación entre procesos en LINUX.

Herramientas necesarias:

Para resolver los ejercicios propuestos necesitará:
1. Una PC con SO Windows con el Software de Virtualización VMWARE.
2. El material proporcionado por la Cátedra.

Fuentes de Información sugeridas

Encontrará información útil en:

- Archivos fuentes de los comandos de LINUX
- Páginas de manual de LINUX
- Guía de Laboratorio IPC de la cátedra + Anexo.

Requisitos de Entrega

Lugar y Fecha de entrega:

1. La fecha de entrega para este práctico será informada por el CAMPUS en el momento de publicar el TP.
2. Los trabajos deben ser entregados vía CAMPUS en 2 archivos, AMBOS con el formato: **"TP3-GRUPO-XX.ZZZ"** (XX es el número que identifica al grupo y ZZZ la extensión del archivo).
3. No se aceptarán trabajos incompletos.

Formato de Entrega.

Deberá enviar dos archivos con la resolución del trabajo:

1. La imagen de un diskette en formato ext2 conteniendo los **programas fuentes** y los **programas compilados**.
2. El segundo, es un archivo de texto. Deberá reunir las siguientes características:

1. Secciones del documento (Todas obligatorias):
   1.1. **Carátula de presentación:** Debe incluir OBLIGATORIAMENTE:
      1.1.1. Asignatura
      1.1.2. Número y Descripción del trabajo práctico
      1.1.3. Año y Cuatrimestre de Cursado
      1.1.4. Identificación del Grupo
      1.1.5. Nombres, Apellidos, LU y direcciones de correo electrónico de TODOS los Integrantes del grupo
   1.2. **Sección Principal:** Aquí debe incluirse la resolución de cada uno de los problemas planteados. **El código de los problemas deberá estar en formato de**

TEXTO (no captura de imagen). Para cada respuesta debe indicarse OBLIGATORIAMENTE, el número y título del problema al que corresponde tal como aparece en el enunciado.

1.3. **Sección de Descargos:** Aquí debe incluirse cualquier comentario que deba tenerse en cuenta para la corrección del práctico. Use esta sección para indicar cosas como:

- Que no pudo resolver alguno de los problemas
- Que no pudo resolver COMPLETAMENTE alguno de los problemas.
- Que no está seguro si el problema está resuelto correctamente.

Comentar los problemas en esta sección es la única forma de obtener puntaje parcial para un ítem que no está bien resuelto. Si se encuentra un problema no resuelto o resuelto de manera INCOMPLETA y eso no está comentado en esta sección, perderá puntos adicionales (no solo le descontaremos puntos por el error sino también por no avisarnos). Si no tiene ningún comentario, deje esta sección en blanco.

Penalizaciones.

Los prácticos entregados en fechas posteriores al límite fijado, tendrán una quita de puntos. Para ver el método empleado para restar puntos consulte la página Web de la Cátedra.

Cambios al enunciado del práctico, fechas de entrega, etc.

Cualquier cambio en los enunciados, fechas de entrega, etc. será informado utilizando dos métodos:

1. La página Web de la Cátedra
2. La lista de correos.

El alumno no puede alegar que no estaba al tanto de los cambios si esos cambios fueron anunciados utilizando alguno de los dos métodos.
SUGERENCIA: Consulte frecuentemente la página de la cátedra y asegúrese de que ha sido incorporado a la lista de correos.

Honestidad académica:

Está bien hablar entre los grupos acerca de cómo resolver problemas, pero los grupos son de 3 integrantes.

No entregue el trabajo de otras personas como propio. Tampoco entregue trabajos publicados en Internet como propios sin citar las fuentes.

Cualquier trabajo, porción de trabajo o texto sin la cita correspondiente es plagio.

Cada grupo debe mantener su código para sí mismo, si su proyecto es copiado, puede ser difícil determinar quién es el verdadero autor.
Cualquier ayuda que reciba deberá documentarla como un comentario al inicio del programa. Por ejemplo, si encuentra una solución a un ejercicio en un texto o manual, debería citar la fuente. Una razonable ayuda, no afectará la aprobación de los trabajos pero fallas al citar las fuentes o la ausencia de las mismas es fraude.

Queda debidamente aclarado, que los trabajos son de autoría, desarrollo y elaboración propia y no de un tercero.

El personal docente de la cátedra se reserva el derecho de tomar coloquio sobre los trabajos prácticos entregados por los alumnos.

MEMORIA COMPARTIDA Y SEMÁFOROS

EJERCICIO 1.

En una pequeña ciudad, dos clubes de barrio han decidido unirse y formar una sola institución. El primer tema en el que estuvieron de acuerdo, fue cómo será la bandera que identificará al nuevo club.

Anteriormente el club 1 tenía su bandera con los colores primarios:

ROJO – AMARILLO – AZUL

y el club 2 con los colores secundarios:

NARANJA – VERDE – VIOLETA

La nueva bandera será intercalando estos colores comenzando por el primer color del club 1 y se completará sucesivamente manteniendo el orden anterior (primario-secundario)

Deberá simular la situación anterior considerando:

- La bandera será representada por un segmento de memoria compartida.
- Cada club será un proceso y se encargará de estampar (imprimir) el color en el orden de su bandera original –cada color será representado por tres letras- en la nueva bandera. Previo a realizar esta acción deberá mostrarlo por salida estándar.
- El estampe alternado de colores será controlado por semáforos.
- Al finalizar se deberá mostrar la nueva bandera generada.

CLIENTE Y SERVIDOR CON MENSAJES

EJERCICIO 2.

Desarrolle un proceso servidor al cual se le pida la fecha y hora utilizando tuberías con nombres (conocidas como named pipes o FIFOs).
El nombre de la FIFO del servidor será "/tmp/datetime.fifo"
Todo proceso cliente creará su propia FIFO con nombre "/tmp/<PID>.fifo"
El servidor estará a la espera de recibir una petición de algún cliente.
Los clientes realizarán su pedido enviando un mensaje a la FIFO del servidor consistente en su PID y esperará la respuesta leyendo su propia FIFO.
Cuando el servidor recibe el PID por su FIFO, obtiene la fecha y hora del sistema. Este es un programa ejemplo de cómo hacerlo:

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <fcntl.h>
```

```
#include <errno.h>

#include <sys/types.h>
#include <sys/stat.h>

main(int argc, char *argv[])
{
       char *dt_ptr;
       time_t t_secs;
       int ret;

       // crea la FIFO
    ret =  mkfifo("/tmp/datetime.fifo", 0666);
       if(ret < 0) {
              fprintf(stderr,"Error %d al crear FIFO\n", errno);
              exit(1);
       }
       // Cambia permisos de la FIFO
       ret =  chmod("/tmp/datetime.fifo", 0666);
       if(ret < 0) {
              fprintf(stderr,"Error %d en chmod\n", errno);
              exit(1);
       }
       // Obtiene los segundos desde 1/1/1970
       t_secs = time(NULL);
       // los convierte a texto
       dt_ptr = ctime(&t_secs);
       printf("Fecha y Hora: %s seg1970: %ld\n",dt_ptr, t_secs);

       // NO HACE NADA POR 20 SEGUNDOS (puede ver si la FIFO esta creada)
       sleep(20);

       // elimina la fifo
       ret = unlink("/tmp/datetime.fifo");
       if(ret < 0) {
              fprintf(stderr,"Error %d al suprimir FIFO\n", errno);
              exit(1);
       }
}
```

El formato de la fecha/hora será:
"Mon Nov 28 23:23:15 2022\n"
Retornado por ctime()
El servidor presentará en pantalla el PID del proceso cliente que le hizo el pedido, junto con la fecha/hora obtenida.
Luego, conformará el nombre de la FIFO del cliente, abrirá la FIFO y escribirá la fecha y hora.
El cliente lee su propia FIFO, presenta en pantalla fecha/hora, cierra y elimina la FIFO.
Algunas de las llamadas al sistema o funciones de librería utilizadas:

- mkfifo
- sprintf
- time
- ctime
- https://www.geeksforgeeks.org/named-pipe-fifo-example-c-program/
- chmod
- unlink

NOTA: Como en las tuberías el tamaño de mensaje es de 1 byte, puede que dos clientes simultáneamente superpongan sus mensajes en la FIFO del servidor, por ejemplo:
PID1: **1234** y PID2: **6789** podría resultar en que se escriba en la fifo "**12**6789**34**" por lo tanto, para escribir en la fijo, lo recomendable es utilizar un semáforo mutex para impedir escrituras simultáneas como muestra el pseudocodigo

```
Down(mutex)
Write(FIFO, PID)
Up(mutex)
```

Para que el semáforo permanezca más allá de los clientes, lo debería crear el servidor.

EJERCICIO 3. IDEM anterior utilizando Colas de Mensajes (System V message queues)

https://www.softprayog.in/programming/interprocess-communication-using-system-v-message-queues-in-linux