Ejercicios

GESTIÓN Y COMUNICACIÓN ENTRE PROCESOS

Objetivo: Afianzar los conocimientos adquiridos sobre creación, gestión y comunicación entre procesos bajo el Sistema Operativo LINUX.

1. Realizar un programa C que realice la creación de un proceso nuevo. Cada proceso tiene asignado como tareas visualizar 2 mensajes, en uno de ellos indica si es el proceso padre o hijo. En el siguiente mensaje muestra su identificador de proceso y también el identificador de proceso de su padre. Cuando no tienen más tareas finalizan.

2. Realice un programa en C que se encarge de crear un nuevo proceso hijo. El programa recibirá como parámetro un número entero que el proceso padre, tras crear al proceso hijo, deberá decrementar en 3 y lo mostrará en pantalla y el proceso hijo lo incrementará en 4 y lo mostrará también en pantalla.

3. Realizar un programa C que realice la creación de 3 procesos hijos, cada proceso visualizará su identificador de proceso y el identificador de su proceso padre y además:

    - El proceso padre definirá una variable denominada dato y que estará inicializada a 5, antes de crear ningún proceso hijo.
    - El proceso hijo 1: se encargará de realizar 5 iteraciones, con 1 segundo de espera entre cada iteración y mostrando en cada una de ellas los ficheros existentes en el directorio actual.
    - El proceso hijo 2: se encargará de realizar tantas iteraciones como indique la variable dato. Entre cada iteración habrá una espera de 2 segundos y visulizará los procesos existentes en el sistema.
    - El proceso hijo 3: realizará tantas iteraciones como indique la variable dato cada 1 segundo y almacenará en cada una de ellas, la palabra hola en un fichero.
    - Seguidamente el proceso principal cambiará dato al valor 3 y mostrará el contenido del fichero creado por el tercer proceso hijo.

4. Realizar un programa C que realice la creación de 4 procesos hijos, cada uno de los procesos escribirá el mensaje “soy el proceso hijo xxx “, en un fichero denominado trabajo.txt, esperará 1 segundo y mostrará en pantalla el mensaje “Finaliza el proceso xxx”, finalizando a continuación, xxx es el identificador de proceso. El proceso padre ejecutará una bucle de 5 iteraciones y en cada una hará una espera de 2 segundos, al salir del bucle mostrara el contenido del fichero.

5. Sobre el programa anterior, realizar la correspondiente modificación que permita que el proceso pader :

   - Espere hasta que termine uno cualquiera de sus hijos.
   - Espere hasta que finalice el proceso 2.

   En ambos casos deberá mostrar la causa por la que ha finalizado el proceso hijo. Realice el programa utilizando las macros explicadas en clase y sin utilizarlas.

6. Realizar un programa C que se encargue de leer la información contenida en dos ficheros, "forden1" y "forden2", cada uno de estos ficheros va a contener 5 números enteros separados por un espacio. El proceso principal se va a encargar de crear dos procesos y cada uno de ellos realizará la ordenación de uno de los ficheros y guardará la información en otro fichero "fordenados". El proceso principal esperará a que terminen ambos procesos hijos, mostrando finalmente el resultado de la ordenación.

7. Realizar un programa C en el que tanto el proceso padre como el proceso hijo escriban sobre un fichero "soy el proceso hijo/padre con identificador xxx". Comentar programa.

8. Realizar un programa C que mediante la creación de un proceso hijo utilizando la familia de llamadas exec, ejecute el comando "ls –l". El proceso principal esperará a la finalización del proceso hijo.

9. Realizar un programa C que implemente la creación de un proceso hijo y se comunique con el mediante la utilización de tuberías (pipe, read). El proceso padre escribirá en la tubería 3 mensajes y el proceso hijo los recibirá y mostrará en pantalla.

10. Realizar un programa C que implemente la creación de un proceso hijo y se comunique con el mediante la utilización de tuberías. ¿ Que sucede si el proceso hijo intenta leer de la tubería sin que el proceso padre haya escrito en ella antes?. Implementar mediante la emisión de señales un sistema que permita leer de la tubería en el momento que otro proceso ha dejado información en ella.

11. Realizar dos programas C, uno de ellos implementará un proceso escritor mediante la utilización de una "FIFO" (MKFIFO), este programa escribe en la fifo el mensaje recogido como parámetro, el otro proceso se encargará de leer de la fifo y mostrar en pantalla los mensajes leidos. Realizar un guión shell que lanze ambos programas, primero escritor y después lector.

12. Realizar un programa C que mediante la utilización de memoria compartida permita la comunicación entre dos procesos, el proceso padre dejará información mensajes en la zona de memoria compartida y el proceso hijo leerá la información y la visualizará.