**Ejercicio 1**

Implementa un programa que lance varios procesos (pasar el número de procesos como argumento) que compartan una zona de memoria compartida. En esa zona de memoria cada proceso deberá escribir su PID diez veces. Proteger la memoria compartida con un semáforo para que no puedan escribir varios procesos a la vez.

**Ejercicio 2**

Implementa dos programas que compartan una zona de memoria compartida. El primero se ejecutará de forma indefinida esperando que el otro escriba algo en memoria compartida para mostrarlo por pantalla (utilizar sleep para no sobrecargar la CPU). El otro recibirá como parámetro un mensaje que deberá escribir en memoria compartida para que el primero lo muestre. Utilizar semáforos para que los procesos compartan de forma ordenada la memoria.

**Ejercicio 3**

Crea un proceso que reciba una señal de tipo SIGALRM cada cinco segundos. Cuando reciba la señal debe notificarlo por pantalla.

**Ejercicio 4**

Crea un programa que tenga cuatro hijos. El programa esperará un número aleatorio de segundos de 1 a 10 y enviará una señal a un hijo, repitiendo el proceso de forma indefinida y tratando el listado de hijos como una cola circular. Los hijos mostrarán su PID y su PPID cada vez que reciban una señal.

**Ejercicio 5**

Crear aplicación multiproceso con un programa servidor que cree 5 procesos hijo. Estos procesos compartirán una zona de memoria compartida en la que almacenarán sus PID en orden de creación (una forma cómoda de hacer esto es que lo haga el proceso padre del resto, es decir, el proceso inicial). Los PID se utilizarán para el envío de la señal SIGUSR1 entre los procesos con el objetivo de que los procesos se intercambien el "turno de activación". Sólo uno de los procesos podrá tener el turno en cada momento. Inicialmente el turno pertenecerá al primer hijo y, posteriormente, lo pasará a otro proceso hijo. El proceso que reciba el turno deberá esperar un tiempo aleatorio entre 1 y 10 segundos antes de pasarlo a otro proceso hijo. El siguiente proceso se elegirá de forma aleatoria. El envío de la señal se protegerá con un semáforo. Este proceso se repetirá un máximo de 100 veces (el contador también puede almacenarse en memoria compartida).

Crear un nuevo programa que enviará un número entre 1 y 5 (los procesos hijos)al otro programa. Si se acierta con el hijo que tiene el turno se mostrará “acierto” por pantalla y nada en caso contrario. Para que no se realice el envío en medio de un cambio de turno se sincronizará el envío con el semáforo que protege el envío de señales en el otro proceso.

Hacer lo mismo pero como multihilo

**Ejercicio 6**

Escribe dos programas que compartan una tubería. El primero se comportará como un servidor, recibiendo los mensajes por la tubería: en el caso de ser una letra se pasará a mayúsculas, y en caso de ser un número se comprobará si es mayor o menor de 10. El segundo programa recibirá como argumento o una letra o un número y lo enviará al primer programa.

**Ejercicio 7**

Implementa el ejercicio anterior con una cola de mensajes.