12. Trabajo Práctico 12 - Procesos y Señales

12.1. Ejercicio 1

Realizar un programa que genere dos procesos, uno padre y uno hijo. El proceso padre debe imprimir la leyenda "Soy el proceso padre, mi PID es ... y el PID de mi hijo es ...". El proceso hijo debe imprimir la leyenda "Soy el proceso hijo, mi PID es ... y el PID de mi padre es ...". Se deben realizar tres versiones de este programa, una donde el proceso hijo finalice **correctamente**, una donde el proceso hijo quede en estado **zombie** y una donde el proceso hijo quede en estado **huérfano**.

Ayuda

Ayudarse de la función **sleep()** para provocar los distintos estados del proceso hijo. Verificar dichos estados con **ps -elf**.

12.2. Ejercicio 2

Realizar un programa que capture la señal SIGINT y que cada vez que la reciba imprima la leyenda "Recibi SIGINT. Presionaste CTRL-C!". Luego de recibir 5 veces la señal, el proceso debe terminar. **Nota: No usar variables globales.**

12.3. Ejercicio 3

Modificar el programa del Ejercicio 2 para que el proceso no pueda terminarse mediante la señal SIGINT (debe seguir imprimiendo la leyenda). Además, el programa debe capturar la señal SIGALRM y programar al SO para la envíe luego de transcurridos 15 segundos. Una vez recibida la señal SIGALRM, debe imprimir en pantalla "SIGINT recibida ... veces" y terminar su ejecución. **Nota: Se puede utilizar una variable global.**

12.4. Ejercicio 4

Realizar un programa que comunique un proceso padre y un proceso hijo a través de las señales SIGUSR1 y SIGUSR2. El padre debe enviar la señal SIGUSR1 al hijo, quien estará esperando recibirla y responderá con la señal SIGUSR2 para luego finalizar su ejecución. Una vez recibida la señal SIGUSR2 por el padre, dicho proceso también finalizará su ejecución.

12.5. Ejercicio 5

Realizar un programa donde se solicitará que se ingrese por consola la cantidad de procesos hijos a crear. Luego el programa generará esa cantidad de procesos hijos. El proceso padre imprimirá "Nuevo hijo! Total de hijos en ejecucion: ..." luego de cada fork() exitoso. Los procesos hijo harán un sleep(1) para luego imprimir "Proceso hijo terminado" y finalizar su ejecución. Para atender cada una de las defunciones de dichos procesos y recoger su status final, se capturará la señal SIGCHLD. El proceso padre simplemente esperará a que todos los procesos hijos terminen su ejecución, imprimiendo la leyenda "Proceso hijo terminado! Total de hijos en ejecucion: ..." luego de recoger el status en el handler de SIGCHLD. **Importante: El programa debe poder soportar la generación de 1000 procesos hijo sin colgarse en su ejecución.**

Ayuda

Revisar la función **waitpid()** y en especial la opción **WNOHANG**.