Session 5: Signals

Objetivo: Gestionar la recepción de signals así como la máscara de bloqueo de signals

Ejercicio 1:

Crea un programa *ej1.c* que imprima un mensaje cada vez que reciba un signal, mostrando el signal recibido.

Ejercicio 2:

Crea un programa *ej2.c* que cree un nuevo proceso que debe contar el número de signals SIGUSR1 recibidos, que mostrará al recibir un SIGUSR2 momento en el que terminará. El programa principal, nada más crear al hijo, debe enviarle 1000 signals SIGUSR1 y 1 SIGUSR2.
Asegúrate de minimizar el número de signals perdidos. ¿Cuántos consigues recibir?

Ejercicio 3:

Crea un programa *ej3.c* que muestra un mensaje cada 2 segundos y muere al recibir un SIGINT, mostrando el número de mensajes mostrados.

Ejercicio 4: Esperar al hijo

Crea un programa *ej4.c* que cree tantos hijos como diga el primer parámetro. A continuación suspenderá su ejecución hasta que algún hijo finalice su ejecución, momento en que se mostrará el pid del hijo finalizado y su resultado.

Ejercicio 5:

Crea un programa *busy.c* que espera un número de microsegundos pasado como parámetro. La espera debe usar la llamada *usleep*. Además queremos medir el tiempo que está esperando, y mostrar la diferencia de tiempo entre lo que le pedimos y lo que realmente espera. Para medir el tiempo puedes usar la llamada *gettimeofday*.

Ejercicio 5b: Haz una nueva versión del programa que haga lo mismo, pero en lugar del *usleep* realice una espera activa. ¿Qué puedes decir sobre los tiempos de espera comparados con el ejercicio anterior?

Ejercicio 5c: Añade a los programas anteriores un parámetro que indique el número de threads a crear, cada uno haciendo lo mismo (esperar N usecs). Haz una gráfica que muestre cómo evolucionan las diferencias de tiempos cuando aumentas el número de threads (especialmente en el punto que haya más threads que cpus/threads físicos tengas en tu máquina).