# Práctica 4: Señales

## Señales como mecanismo de sincronización

Las señales pueden utilizarse como un mecanismo rudimentario de comunicación entre procesos. Cuando un proceso solicita al kernel que envíe una señal a otro proceso (con la llamada `kill()` o con `siqqueue()`) se esta produciendo un intercambio de información entre los dos procesos que aunque restringido puede ser suficiente para sincronizar procesos.

Para ilustrar este uso, en esta práctica vamos a comunicar a dos procesos emparentados (padre e hijo) mediante un fichero regular que servirá de buzón (fichero `mailbox`) siguiendo el siguiente protocolo:

- El proceso padre deberá leer un mensaje (tamaño máximo MAXMSJ) por la entrada estándar que comunicará al hijo a través de dicho fichero `mailbox`.
- La forma en la que el proceso padre le indicará al hijo que hay datos disponibles en `mailbox` es mediante el envio de una señal (SIGUSR1).
- El hijo, cuando haya efectuado la lectura, le enviará otra señal (SIGUSR2) al padre para indicarle que esta listo para recibir más mensajes.
- Una vez que el proceso padre haya recibido la señal SIGUSR2, volverá a leer una nueva cadena por la entrada estándar tras una pausa de los segundos que indique la variable de entorno PAUSASECS y se repetirá de nuevo la comunicación. Si PAUSASECS no esta definida por defecto se hara una pausa de 1 segundo.

Este protocolo de comunicación se interrumpirá cuando el mensaje leido por el proceso padre contiene solo el carácter fin de fichero (EOF/CTRL+D). En este caso el padre enviará al hijo la señal SIGTERM y terminará su ejecución.

Para el acceso a la variable de entorno PAUSASEC se utilizará la llamada `getenv()` (consultar su uso). Para el acceso al fichero mailbox y la lectura de la entrada estandar se pueden utilizar funciones de la librería de C (`fopen`, `fgets`, `fputs`).