Práctica 3: **Llamadas a Sistema**

Abril 2020

Índice

1. Introducción

El objetivo de esta práctica se centra en el uso del lenguaje C y la utilización de las llamadas a sistema. El objetivo es la implementación de un temporizador digital distribuido entre múltiples procesos. La comunicación entre procesos se realizará mediante señales y tuberías.

2. Restricción en el uso de funciones C

Para realizar esta práctica es obligatorio utilizar las llamadas a sistema de Unix. No se podrán utilizar funciones C de entrada/salida (*scanf*, *printf*, *fopen*, etc.). En su lugar habrá que utilizar las llamadas a sistema (*open*, *read*, *write*, *close*, ...). Sí que esta permitido utilizar funciones C para formatear y manipular cadenas (*sprintf*, *strlen*, ...). Para ello se incluye un ejemplo de uso, ver código `sprintf.c`. Tampoco está permitido utilizar la función *sleep*, sino que en su lugar habrá que utilizar las funciones *pause* y *alarm* cuando sea necesario.

3. Código a implementar

Se desea implementar un temporizador digital mediante 4 procesos independientes que gestionaran los segundos, los minutos y las horas. La funcionalidad de cada proceso es la siguiente; los detalles se describen a lo largo del enunciado.

1. El proceso principal gestionará el reloj. Este proceso es el proceso padre de los tres procesos hijo que se describen a continuación.

2. El primer proceso hijo, que denominaremos coloquialmente "proceso segundos", contará los segundos que transcurren.

3. El segundo proceso hijo, que denominaremos coloquialmente "proceso minutos", contará los minutos que transcurren.

4. El tercer proceso hijo, que denominaremos coloquialmente "proceso horas", contará las horas que transcurren.

El funcionamiento de cada uno de los procesos es el siguiente.

1. Se ejecutará únicamente el proceso principal el cual creará los tres procesos hijo descritos anteriormente. Los contadores de los tres procesos hijos se inicializaran a cero. El proceso principal se comunicará con sus procesos hijos utilizando tuberías. En particular, se recomienda que se cree una tubería independiente para cada proceso hijo. También es conveniente que el proceso principal imprima por pantalla, al comenzar, su identificador de proceso para que se puedan realizar experimentos de forma sencilla.

2. Cuando el proceso principal reciba por primera vez la señal SIGUSR1 (que se enviará desde otro terminal mediante el comando `kill`) se iniciará el temporizador digital. Para ello el proceso principal mostrará un mensaje por la salida estándar. El proceso principal enviará una señal SIGUSR1 al proceso segundos para indicarle que se inicia el contador.

3. El proceso segundos utilizará la función *alarm* para sea avisado cada vez que pase un segundo. Cuando hayan pasado 60 segundos, el proceso segundos reiniciará su contador a cero y enviará una señal SIGUSR2 al proceso minutos. Para poder enviar esta señal el proceso segundos necesitará saber el identificador (PID) del proceso minutos. El proceso principal se lo tendrá que haber "indicado" antes (hay diferentes formas de implementarlo, utilizad la que creáis conveniente).

4. El proceso minutos incrementará su contador cada vez que reciba una señal SIGUSR2. Cuando el proceso minutos haya recibido 60 veces la señal SIGURS2 reiniciará su contador a cero y enviará una señal SIGURS2 al proceso horas. De forma similar al proceso segundos, el proceso minutos necesitará saber el identificador (PID) del proceso horas. El proceso principal se lo tendrá que haber "indicado" antes (para ello utilizad la forma equivalente que habéis utilizado para que el proceso principal le indique al proceso segundos el PID del proceso minutos).

5. El proceso horas incrementará su contador cada vez que reciba una señal SIGUSR2. Este contador se irá incrementado indefinidamente.

6. El proceso principal es el que permitirá imprimir el valor de los contadores de los hijos. Cada vez éste reciba una señal SIGUSR1 (que se enviará desde otro terminal mediante el comando `kill`), el proceso principal enviará una señal SIGUSR1 a sus procesos hijos (segundos, minutos, horas). Los procesos segundos, minutos y horas escribirán en sus tuberías los valores de sus contadores (se aconseja que sea en formato binario, ver código `fork-pipe.c`). El proceso principal, por su parte, leerá de las tuberías los valores e imprimirá por pantalla el contador digital en formato "hh:mm:ss". Para formatear la cadena en este formato se recomienda utilizar la función *sprintf*. En concreto, se incluye un ejemplo junto con esta práctica.

7. Cuando el proceso principal reciba la señal SIGTERM se parará el temporizador digital: para ello el proceso "principal" enviará primero una señal SIGTERM a el resto procesos (segundos, minutos y horas). Estos tres procesos finalizaran de forma "limpia" (de su función asociada) sin emitir mensaje alguno por pantalla. Una vez realizado enviado el SIGTERM, el proceso principal imprimirá un mensaje y finalizará también de forma limpia. No hace falta que el proceso principal espere que los tres procesos hijo finalicen.

4. Entrega

El único proceso que imprimirá mensajes por la pantalla es el proceso principal. Los procesos segundos, minutos y horas no imprimirán ningún mensaje por pantalla a menos que se trate de un mensaje de error, en cuyo caso se imprimirá el mensaje por la salida de error. En caso que se produzca un error no es necesario gestionar que el resto de procesos finalice.

Para la entrega de la práctica se pide entregar el código fuente (80 %) y un informe (20 %).

4.1. Código fuente

Se pide entregar el siguiente código fuente

- Entregar el código fuente que implemente la funcionalidad descrita en la sección 3. Todo el código se puede implementar en un único fichero C. También se pueden utilizar múltiples ficheros C si así se desea.

- Entregar un `Makefile` que permita compilar el código. La estructura de estos ficheros está descrita en un documento del campus, en la sección de problemas.

4.2. Informe

El informe a entregar debe estar en formato PDF o equivalente (no se admiten formatos como odt, docx, ...). Este informe debe mostrar las pruebas que se han realizado para asegurar el buen funcionamiento del código. Podéis probar de enviar manualmente con el comando `kill`, por ejemplo, señales SIGUSR2 a los procesos minutos y horas para comprobar que los contadores correspondientes se incrementan. No incluir una descripción del algoritmo implementado salvo que creáis relevante comentar algún detalle (como, por ejemplo, cómo le "indica" el proceso principal cuál es el PID del proceso minutos y horas al proceso segundos y minutos respectivamente). Sed breves y claros en los comentarios y experimentos realizados, no hace falta que os extingáis en el texto escrito.

En caso de que queráis incluir capturas de pantalla en vez de incluir los resultados de los experimentos en formato texto, aseguraos de que el texto de la captura se puede leer bien (es decir, que tenga un tamaño similar al resto del texto del documento) y que todas las capturas sean uniformes (es decir, que todas las capturas tengan el mismo tamaño de texto).

El documento debe tener una longitud máxima de 2 páginas (sin incluir la portada). El documento se evaluará con los siguientes pesos: pruebas realizadas y comentarios asociados, un 60 %; escritura sin faltas de ortografía y/o expresión, un 20 %; paginación del documento hecha de forma limpia y uniforme, un 20 %.