Miguel Ángel Sánchez
Juan Velasco

miguelangel.sanchezm@estudiante.uam.es
juan.velascoi@estudiante.uam.es

Práctica 2: Memoria

Date: 05/04/18

Ejercicio 2

En este programa el proceso padre crea 4 hijos y cada uno de ellos imprime el mensaje "soy el proceso hijo <PID>". Como después de imprimir este mensaje tienen que dormir 30 segundos, no les da tiempo a imprimir el mensaje de "soy el proceso hijo x y ya me toca terminar" porque el padre los termina (mandándoles la señal SIGTERM) únicamente 5 segundos después de crearlos.

Esta es una captura de pantalla del resultado al ejecutar este ejercicio:

```
miguel_linux@Miguel-Lenovo-G50-70 ~/Escritorio/Infomates_UAM/Segundo cuatri/Pr
acticasSoper/Practica2definitivo $ ./Ejercicio2
Soy el proceso hijo 6462
Soy el proceso hijo 6463
Soy el proceso hijo 6464
Soy el proceso hijo 6465
miguel_linux@Miguel-Lenovo-G50-70 ~/Escritorio/Infomates_UAM/Segundo cuatri/Pr
```

Ejercicio 4

En este ejercicio el padre crea procesos hijos que van haciendo un trabajo. Cuando terminan de hacer este trabajo mandan una señal al padre para que cree otro hijo y es este nuevo hijo el que termina al anterior. El último hijo será terminado por el padre que después acabará.

Para este ejercicio hemos utilizado 2 señales: SIGUSR1 es la que le mandan los hijos al padre para que salga del pause y cree un nuevo hijo. Y SIGUSR2 es la señal que mandará este nuevo hijo al hijo anterior, que hará que ese hijo termine.

Ejercicio 6

En este ejercicio se nos pedía modificar un código dado para conseguir una salida parecida de dos formas distintas:

- **Apartado a**

  El proceso padre crea un hijo y lo espera.

  El hijo establece una alarma con un temporizador de 40 segundos nada más ser creado. A pesar de haberla bloqueado antes, cuando el hijo recibe la señal de alarma, está desbloqueada, por lo que el proceso termina su ejecución. Si no la hubiésemos desbloqueado no habría tenido efecto la señal de alarma.

- **Apartado b**

  Este programa a pesar de estar implementado de forma diferente al apartado a, tiene una salida muy parecida, puesto que el hijo está vivo durante 40 segundos y después termina. La diferencia es que en el apartado a lo termina la señal SIGALARM que él mismo ha establecido, y en el apartado b lo termina el

padre mandándole SIGTERM 40 segundos después de crearlo. Además, en este apartado el hijo imprime un mensaje antes de finalizar.

Ejercicio7libreria

Este ejercicio es una adaptación que hemos hecho del ejercicio7 dado en la práctica. Es un tester que comprueba las funciones creadas en nuestra librería semáforos.c. Para probar bien todas las funciones hemos añadido comprobaciones de UpMultiple y DownMultiple, que no venían incluidas.

Ejercicio 8

De este ejercicio no hay mucho que comentar. Hemos creado todas las funciones requeridas en el enunciado en los ficheros semaforos.h y semaforos.c. Luego para usar nuestras nuevas funciones de semáforos simplemente habrá que hacer un "include" de la librería semaforos.h. Esta librería ha sido probada en el Ejercicio7libreria y también se utiliza en el Ejercicio9.

Ejercicio 9

En este ejercicio el proceso padre crea N hijos que son los cajeros. Estos cajeros tienen que leer de un fichero de clientes diferentes sumas de dinero e irlas sumando a sus respectivas cajas. Cuando estas cajas superen los 1000 euros el cajero mandará una señal al padre para que, cuando pueda, retire 900 euros de su caja y los meta en una cuenta global (que hemos puesto en otro fichero), mientras tanto el cajero seguirá cobrando a sus clientes. Por último, cuando en una caja ya no quedan mas clientes, el cajero manda otra señal al padre que, otra vez cuando pueda, retirará el dinero restante de la caja y lo meterá en la cuenta global.

Para la realización de este ejercicio es necesario utilizar dos señales: SIGUSR1 la hemos asociado a la función de MasDe1000, o sea que la enviará el hijo al padre cuando en su caja haya mas de 1000 euros. Mientras que SIGUSR2 la hemos asociado a la función de ClientesAcabados, esta señal la mandará el hijo cuando haya acabado de leer el archivo de clientes.

Para proteger los ficheros en los que se guarda el dinero de las cajas (puesto que son ficheros en los que escriben y leen tanto los cajeros como el proceso padre) es necesario utilizar semáforos. Hemos utilizado un semáforo para cada caja, por lo que antes de leer o escribir en una caja será necesario hacer Down de ese semáforo y justo después habrá que hacer Up.

Pero el proceso padre necesita saber de alguna forma quién es el proceso que le está mandando la señal que ha recibido. Es por ello que hemos creado el fichero turnos.txt en el que se imprime la caja que manda la señal. Este fichero lo hemos protegido también con un semáforo puesto que todos los hijos pueden escribir en él y podría haber sobre escrituras sin que el padre llegase a leer el turno. Este semáforo lo bajará el proceso hijo justo antes de imprimir su turno en el fichero, y lo subirá el proceso padre después de hacerse cargo de la señal recibida.

Nuestro ejercicio funciona correctamente en la mayoría de las ocasiones. Lamentablemente, hay veces en las que sin saber por qué, justo cuando está a punto de terminar el programa, el proceso padre lee del fichero de turnos un número altísimo que nadie ha escrito. En consecuencia, intenta acceder a una caja que no existe y a partir de ahí el programa falla. Tras horas buscando no hemos conseguido averiguar dónde está el fallo, pero hemos decidido incluir el ejercicio con esta explicación para aclarar que entendemos el funcionamiento de los semáforos y las señales, pero simplemente no hemos podido encontrar este fallo.

Este es un ejemplo de cuando el programa no termina correctamente:

```
ESTOY HACIENDO LA SENAL DE MASDE1000 en 0
Voy a bajar el semaforo de la caja 0
Ya tengo menos de 1000 en la caja 0
YA HEMOS SACADO 900 EN 0
HE TERMINADO DE HACER LA SENAL DE MASDE1000 en 0
YA HEMOS SACADO TODO EL DINERO EN 0
He impreso el turno 2
MANDO SENAL DE CLIENTES ACABADOS en la caja 2
HE TERMINADO DE HACER LA SENAL DE CLIENTESACABADOS en 0
He leido el turno 32695
ESTOY HACIENDO LA SENAL DE CLIENTESACABADOS en 32695
Voy a bajar el semaforo de la caja 32695
Error al bajar el semaforo 32695
```

Como podemos observar (teniendo en cuenta cuando se hacen estas impresiones indicativas), en el archivo de turnos se imprime un 2, pero luego el proceso padre lee 32695. Esto hace que luego quiera acceder a esta caja y por tanto bajar este semáforo que no existe y ya todo empieza a fallar.

A continuación, se muestra una captura de nuestro ejercicio funcionando correctamente con 2 cajas:

```
En la 1 caja hay 0
YA HEMOS SACADO TODO EL DINERO EN 1
HE TERMINADO DE HACER LA SENAL DE CLIENTESACABADOS en 1
He impreso el turno 0
MANDO SENAL DE NOS HEMOS PASADO DE 1000 en la caja 0
CLIENTES ACABADOS en la caja 0
He leido el turno 0
ESTOY HACIENDO LA SENAL DE MASDE1000 en 0
Voy a bajar el semaforo de la caja 0
En la 0 caja hay 349
YA HEMOS SACADO 900 EN 0
HE TERMINADO DE HACER LA SENAL DE MASDE1000 en 0
He impreso el turno 0
MANDO SENAL DE CLIENTES ACABADOS en la caja 0
He leido el turno 0
ESTOY HACIENDO LA SENAL DE CLIENTESACABADOS en 0
Voy a bajar el semaforo de la caja 0
En la 0 caja hay 0
YA HEMOS SACADO TODO EL DINERO EN 0
HE TERMINADO DE HACER LA SENAL DE CLIENTESACABADOS en 0

Todo ha ido correctamente
```

Y a continuación una captura con 3 cajas:

```
En la 0 caja hay 0
YA HEMOS SACADO TODO EL DINERO EN 0
CLIENTES ACABADOS en la caja 1
HE TERMINADO DE HACER LA SENAL DE CLIENTESACABADOS en 0
He impreso el turno 1
MANDO SENAL DE CLIENTES ACABADOS en la caja 1
He leido el turno 1
ESTOY HACIENDO LA SENAL DE CLIENTESACABADOS en 1
Voy a bajar el semaforo de la caja 1
En la 1 caja hay 0
YA HEMOS SACADO TODO EL DINERO EN 1
HE TERMINADO DE HACER LA SENAL DE CLIENTESACABADOS en 1
NOS HEMOS PASADO DE 1000 en la caja 2
He impreso el turno 2
MANDO SENAL DE NOS HEMOS PASADO DE 1000 en la caja 2
He leido el turno 2
ESTOY HACIENDO LA SENAL DE MASDE1000 en 2
Voy a bajar el semaforo de la caja 2
En la 2 caja hay 147
YA HEMOS SACADO 900 EN 2
HE TERMINADO DE HACER LA SENAL DE MASDE1000 en 2
CLIENTES ACABADOS en la caja 2
He impreso el turno 2
MANDO SENAL DE CLIENTES ACABADOS en la caja 2
He leido el turno 2
ESTOY HACIENDO LA SENAL DE CLIENTESACABADOS en 2
Voy a bajar el semaforo de la caja 2
En la 2 caja hay 0
YA HEMOS SACADO TODO EL DINERO EN 2
HE TERMINADO DE HACER LA SENAL DE CLIENTESACABADOS en 2

Todo ha ido correctamente
```

Como podemos ver, en ambas capturas todas las cajas quedan a cero (esto lo sabemos porque vemos impreso “he terminado de hacer la señal de clientes acabados en x”) y todos los procesos hijos terminan correctamente al igual que el padre.

Hemos decidido incluir una solución alternativa que es el *ejercicio9b.c*. Viendo que cuando el ejercicio fallaba era siempre al final, hemos probado a cambiar la función de clientes acabados que ahora no será la que vacíe la caja sino simplemente aumentará una variable global “acabados”. Cuando todos los procesos hayan acabado el mismo proceso padre mediante el bucle del final vaciará todas las cajas.

Hemos pensado que esta variante también sería válida puesto que en el enunciado se pide que sea el padre quien vacíe las cajas, pero no dice que deba hacerse en la función de manejador de la señal.