Memoria de la práctica 3
Sistemas operativos 2018-2019

Alejandro Pascual y Víctor Yrazusta

11 de abril de 2019

Respuestas a las preguntas cortas

Ejercicio 2

Explica en qué falla el planteamiento del ejercicio

El programa falla porque el incremento del id anterior se produce antes del scanf(), que es donde los procesos se quedan esperando. Esto hace que tras introducir el nombre de un usuario multitud de procesos hayan modificado dicho valor, haciendo que no se corresponda con el valor real.

Detalles de implementación

Ejercicio 2

La versión original la hemos implementado tal y como viene especificada en el usuario, provocando el fallo comentado anteriormente.

Para la versión corregida hemos implementado un semáforo para controlar la sección crítica. Esta sección comienza cuando un proceso hijo sale del sleep() y finaliza cuando el proceso padre ha impreso la información del nuevo usuario. De esta manera todas las entradas se producen de forma secuencial y sin incoherencias.

Ejercicio 3

Para resolver el problema del productor-consumidor en este ejercicio hemos utilizado tres semáforos. El primero es un semáforo binario que gestiona la zona crítica, que en ambos programas representa todo el proceso de producción o consumición. Los otros dos son semáforos n-arios. Uno representa el número de huecos libres, se inicializa al tamaño de la cola (10), se decrementa antes de la producción y se incrementa tras la consumición. El otro representa el número de elementos consumibles, se decrementa antes de la consumición y se incrementa tras la producción.

Hemos observado que los elementos que eran insertados en la cola, situada en la memoria compartida, antes de que el consumidor se ejecutase no podían ser leídos correctamente por este. Debido a este problema inicializamos el semáforo que representa los huecos libres al ejecutar el consumidor.

Ejercicio 4

Para resolver el ejercicio 4 hemos implementado los procesos A, B y C como se especificaba en el enunciado. Hemos usado la llamada a execlp porque era la que más cómoda nos resultaba para ejecutar los procesos después de los fork del padre. Hemos reservado un byte adicional en el mensaje que transmitimos para poder imprimirlo directamente insertando un carácter de terminación de cadena.

El padre espera a la terminación de los hijos y se preocupa de eliminar las colas de mensajes antes de terminar su ejecución.