Memoria Practica (I)

Programación de Servicios y Procesos

Andreu Sanz Sanz

Tabla de contenidos

Que

Realiza un programa en C que cree un proceso (tendremos 2 procesos uno padre y otro hijo).
El programa definirá una variable entera y le dará el valor 6. El proceso padre incrementará dicho valor en 5. El hijo restará 5. Se deben mostrar los valores en pantalla. La salida debería ser algo así:

```
Valor inicial de la variable: 6
Valor en proceso hijo: 1
Valor en proceso padre: 11
```

¿Por qué da este resultado?

Para que

Para tener un primer contacto con C, ver cómo son los procesos, así como su creación y funcionalidad. Además, se utiliza la función fork(), que crea una copia exacta del mismo proceso.

Como

Primero, he incluido las bibliotecas necesarias para esta práctica. Luego, declaro una variable que me servirá para guardar el valor que me devuelva la función fork(), y también declaro una variable que será la que se sumará o restará dependiendo del proceso, le doy el valor de 6 e imprimo su valor. Después, llamo a la función fork() y guardo el valor que devuelve en la variable anteriormente. Podría haber declarado la variable en esta misma línea e guardar el valor de la función fork(, pero personalmente creo que el código queda mucho más limpio si todas las variables quedan definidas en un mismo lugar.

Más tarde, siguiendo las posibilidades de devolución de la función fork(), he creado un algoritmo para saber si el proceso actual es el padre, el hijo o si hay un error. Primero, compruebo si la variable 'pid', donde guardo el valor de fork() es igual a 0, si es así, sabré que este es el proceso hijo, por lo tanto, le resto 5. Después, compruebo si la variable es igual a -1 para saber si ha ocurrido un error. Si es así, imprimo que ha ocurrido un error o de lo contrario, sé que es el proceso padre. En el proceso padre, he añadido una función para que espere hasta que su proceso hijo acabe, de modo que en la salida se pueda ver primero el proceso hijo y luego el padre. Después, incremento la variable en 5 y muestro su valor.

Pseudocodigo

```
Incluir las bibliotecas <unistd.h> y <stdio.h>
Función principal:
    Declarar una variable de tipo pid_t llamada pid
    Declarar una variable de tipo entero llamada variable e inicializarla con 6
    Imprimir "Valor inicial de la variable: " + variable
    Llamar a la función fork() y guardar el resultado en la variable pid
    Si pid es igual a 0 Entonces
        Decrementar pid en 5
        Imprimir "Valor en proceso hijo: " + variable
    Sino Si pid es igual a -1 Entonces
        Imprimir "***Error en el proceso"
    Sino
        Esperar a que el proceso hijo termine con la función wait(null)
        Incrementar pid en 5
        Imprimir "Valor en proceso padre: " + variable
    Fin Si
Fin de la Función principal
```

Conclusión

Creo que es una buena manera de tomar contacto con el tema del multiproceso, aunque debo reconocer que no tengo muchos conocimientos de C. En el curso anterior dimos Java, y hacer esto en C... ¡debo practicar! Es interesante cambiar de lenguaje y ver la manera de programar en otros lenguajes, pero la verdad es que debería investigar para saber cómo se hace esta misma actividad en Java.