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.

Porque:

Porque cuando hemos de programas servicios no nos podemos permitir que nuestro “demonio” se quede suspendido a la espera de una respuesta, por ejemplo de una petición a una base de datos, mientras en la cola de entrada se agolpan peticiones que, con suerte, se podrían resolver de forma rápida. Estos procedimientos nos permiten tratar, en un proceso hijo, esas peticiones sin dejar al servicio, padre, suspendido.

Para que:

Para aprender a utilizar varios procesos simultáneamente y poder dividir nuestro trabajo en porciones mas pequeñas que, sin perder la sincronización requerida, se ejecuten de forma independiente y asi evitar que nuestro sistema, o servicio, pierda disponibilidad y aumente su eficiencia.

Como:

En primer lugar hay que hacer notar que, aunque este ejercicio podría hacerse utilizando un solo proceso, utilizaremos dos procesos por que el objetivo de la practica lo que pretendemos es familiarizarnos con el uso de multiproceso.

Lo primero que hemos hecho es ver que partes del programa son comunes, las cuales realizaremos antes de dividir el proceso en dos, y cuales independientes entre si.

Una vez hecho esto hemos utilizado la función fork para crear ese proceso hijo. No hay que olvidarse de comprobar que esa función no devuelva el valor -1, que indica que el proceso ha fallado y no tenemos ese segundo proceso. He decidido comparar con -1 y no usar otro tipo de comparación (como por ejemplo <0) por que no tengo ninguna documentación que me diga que otros valores negativos signifiquen que ha ido algo mal y tampoco tengo la suficiente experiencia con el muso de la función como para suponerlo. Además si mi código va ha ser revisado por otro programador, que tampoco tiene que ser mas experto que yo en la materia, le facilito el entendimiento y la búsqueda de apartados concretos en el.

Una vez hecha esta comprobación he decidido, mediante un if, bifurcar el programa en la parte del padre, proceso original, y el hijo, proceso nevo creado por el fork. Como nuevamente la especificación me dice que en el hijo la función fork devolverá 0, utilizo este valor como referencia por las mismas razones que en el caso anterior.

Podría haber utilizado un switch para unificar los dos apartados anteriores, pero siendo mi primer programa he preferido utilizar un if por que me parece mas legible.

El resto del código en C++ estándar y pienso que al nivel de 2º de DAM no merece explicación. Tan solo comentar que la utilización de la función wait para evitar que el padre termine antes que el hijo. Aunque en este caso no creo que diera problemas al no utilizar el padre los resultados del hijo.

Conclusión:

Creo que es una bueno manera de tomar contacto con el tema del multiproceso, aunque he de reconoces que mi C esta algo oxidado. En el curso anterior nos centramos en Java y cuesta un poco acostumbrarse. Como programador me parece interesante salir del mundo Java, como alumno... ¿Se podría hacer esto en Java?