**Solución Laboratorio 3 S.O**

**1.**

* **¿Cuál es el valor de la variable en el proceso hijo?**

Si se realiza la asignación de la variable antes de hacer el fork y no se modifica después, el valor de la variable se mantiene con el valor que fue declarada

* **¿Qué sucede con la variable cuando el proceso hijo y el padre cambian el valor de x?**

Adquieren valores diferentes en cada uno, es decir, la variable en le proceso hijo no le afecta los cambios en el proceso padre y viceversa.

**2.**

* **¿Pueden el padre y el hijo acceder al file descriptor retornado por open()?**

Si

* **¿Qué pasa si ellos empiezan a escribir el archivo de manera concurrente, es decir, a la misma vez?**

Primero termina el padre de escribir y despues termina el hijo.

**3.**

* **¿Usted podría hacer esto sin llamar wait() en el padre?**

Si, con un sleep de un segundo o con un waitpid.

**4.**

* **¿Por qué piensa usted que existen tantas variaciones para la misma llamada básica?**

En esencia todas hacen los mismo, sin embargo, se cada una difiere en que las instrucciones del proceso que las invoca son sustituidas por las instrucciones del proceso indicado

**5.**

* **¿Cuál es el valor de retorno de la función wait()?**

La función wait retorna el PID del hijo (si se usa en el padre)

* **¿Qué pasa si usted usa la función wait en el hijo?**

Si la función wait se usa en el hijo, el retorno del wait es -1, lo que significa que hay un error o no es necesario hacer un wait

**6.**

* **¿Cuándo podría ser waitpid() útil?**

Suspende la ejecución del proceso en curso hasta que un hijo especificado por el argumento pid ha terminado, o hasta que se produce una señal cuya acción es finalizar el proceso actual o llamar a la función manejadora de la señal

**7.**

* **¿Qué pasa si el hijo llama printf() para imprimir alguna salida después de cerrar el descriptor?**

El hijo no imprime nada, pues al utilizar close hace que no se ejecute el proceso de printf