**MEMORIA PRACTICA 1 SISTEMAS OPERATIVOS 2016-2017**

**Ejercicio 3:**

En este ejercicio realizamos las mismas operaciones con un proceso hijo y con un hilo para comprobar si hay una diferencia de tiempos.

Para medir la diferencia de tiempos utilizaremos la estructura timeval y la función gettimeofday(). Tras ejecutar los apartados a y b comprobamos que los hilos son más rapidos (en torno a 10 segundos para una muestra de 10000 primos):





**Ejercicio 4:**

En este ejercicio realizaremos un programa con dos hilos que comparten datos entre ellos. Para ello utilizaremos la siguiente estructura:

struct \_Hilo

{

Hilo \*otro; /\*!<Estructura del otro hilo\*/

int progreso; /\*!<Progreso del cálculo\*/

int numero; /\*!<Número del hilo\*/

int mul; /\*!<Multiplicador aplicado a el hilo\*/

pthread\_t p; /\*!<El hilo\*/

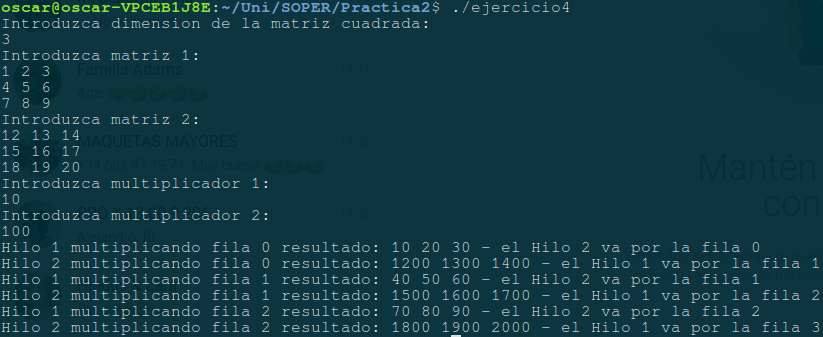
int \*\*matriz; /\*!<matriz a calcular\*/

int \*dim; /\*!<Dimensión de la matriz\*/

};

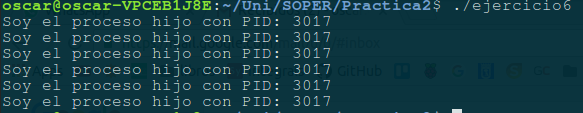
En ella almacenamos la estructura del otro hilo y los datos necesarios del hilo, como el progreso de la multiplicación de matrices, la matriz original y la dimensión de la matriz.

Gracias a esta estructura podemos realizar el cálculo:



**Ejercicio 6:**

En este ejercicio utilizaremos una señal para detener un hijo. Para ello guardaremos el pid del hijo en una variable en el padre y dividiremos el código en un bucle del hijo que cada 5 segundos imprima por consola. Cuando pasan 30 segundos, el padre “matará” al proceso hijo con la señal SIGKILL y terminará.



**Ejercicio 10:**

En este ejercicio decidimos implementar una máscara para capturar señales. El proceso A tiene una sección crítica (Escritura de fichero), por lo que, como el proceso B tiene que leerlo, paramos el proceso B hasta que A termine su sección crítica para darle paso a B. Esto lo realizamos por medio de las señales USR1 y USR2.

Para comprobar si A está activo utilizamos waitpid con la opción WNOHANG, para que el padre no se bloquee y continue con la ejecución.