Actividad 4

1. Ejer1:

```
labsmac@ubuntuarm:~$ nano ejer1.c
labsmac@ubuntuarm:~$ gcc -o ejer1 ejer1.c
labsmac@ubuntuarm:~$ ./ejer1
Soy el proceso padre con PID 2623 y he creado un hijo con PID 2624
Soy el proceso hijo con PID 2624 y voy a terminar al proceso padre con PID 2623
Killed
```

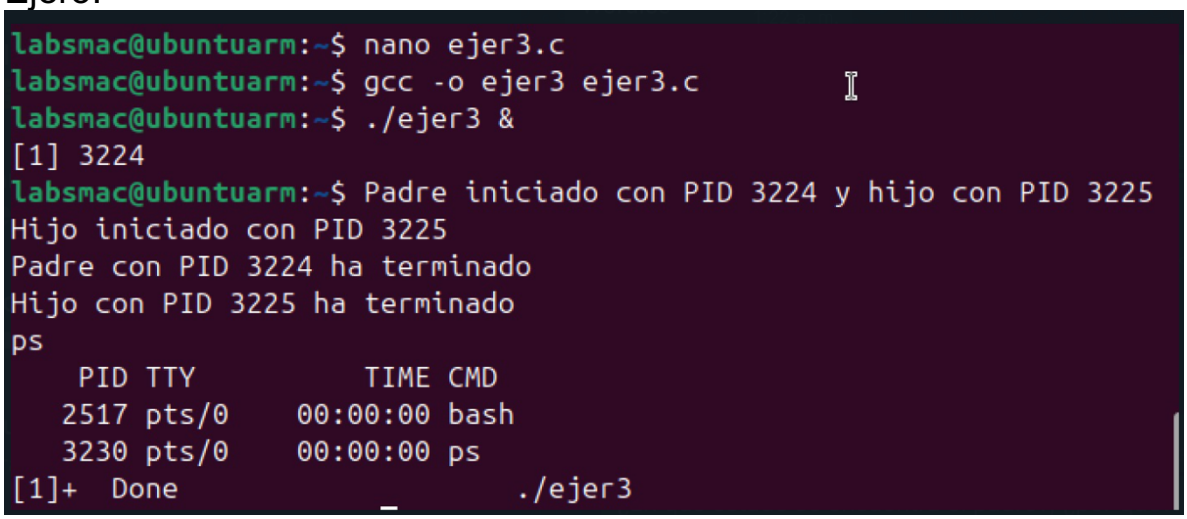
2. Ejer2:

```
labsmac@ubuntuarm:~$ gcc -o ejer2 ejer2.c
labsmac@ubuntuarm:~$ ./ejer2 &
[1] 3146
labsmac@ubuntuarm:~$ Padre iniciado con PID 3146 y hijo con PID 3147
Hijo iniciado con PID 3147
Hijo con PID 3147 ha terminado
ps
    PID TTY          TIME CMD
   2517 pts/0    00:00:00 bash
   3146 pts/0    00:00:00 ejer2
   3147 pts/0    00:00:00 ejer2
   3148 pts/0    00:00:00 ps
labsmac@ubuntuarm:~$ Padre con PID 3146 ha terminado
ps
    PID TTY          TIME CMD
   2517 pts/0    00:00:00 bash
   3150 pts/0    00:00:00 ps
[1]+  Done                    ./ejer2
```

¿En qué estado se encuentra el hijo?, ¿por qué?:

El proceso hijo tendra una duracion de un segundo, por lo tanto, cuando se ejecuta 'ps' despues de ese timepo, el hijo ya termino. Si este ya termino, se encontrara en <defunct>. Esto pasa por que el proceso hijo termina antes que el padre y , hasta que el padre no llame a wait(), el sistema continua guardando informacion del proceso hijo para permitir que el padre recoja su estado de salida.

3. Ejer3:



¿En qué estado se encuentra el padre?, ¿por qué?:
El padre se encuentra en estado Terminated una vez que termina su ejecucion de 1 segundo, mientras que el hijo sigue en ejecucion. Despues de un segundo el padre termina y el hijo sigue corriendo durante 20 segundos.

4. Ejer4:

El programa me genera multiples procesos padres en lugar de uno solo, lo cual no es el comportamiento que se busca. Tal vez ‘ fork() ‘ se esta creando procesos en lugares inesperados.

Ejecucion del programa

Terminal:

Padre con PID 3109
Padre con PID 3110
Padre con PID 3112
Padre con PID 3111

killed

5. Preguntas:

   A. Si ejecuto un proceso que cree otros procesos, ¿Cómo puedo ver que procesos se están ejecutando?

Se puede usar el comando ‘ps’ en la terminal para listar los procesos actuales. Para ver procesos mas especificos como hijos y nietos puedes usar ‘ps -ef | [nombre del proceso].

B. De los ejercicios 2 y 3, ¿En qué casos hubo procesos zombies?

En el ejericico 2 hubo uno por que el padre termino antes que el hijo. El zombie se creo cuando el hijo termino y el padre aun no habia recolectado su estado.

En el ejercicio 3 no hubo, ya que el hijo duro mas que el padre y al termina, todos los recursos fueron liberados.

C. ¿Qué es un proceso zombie?, ¿conviene evitarlos?, ¿por qué?

Un proceso zombie es un proceso que ha terminado, pero su entrada en la tabla dde procesos sigue presente por que su padre aun no ha llamado a ‘wait()’ para recolectar si estado.

D. ¿Qué aprendiste?

Aprendi a manejrar la creacion de procesos en c y como controlar la duracion y terminacion de procesos padres, hijos y nietos. Tambien la importancia de recolectar procesos para evitar la acumulacion de zombie, lo cual es crucial para el buen rendimiento del sistema.