

# Sistemas operativos

## Practica 1

Gestión de procesos y archivos

Grado en ingeniería informática

Francisco Javier Pérez Martínez 74384305M

Francisco Joaquín Murcia Gómez 48734281H

Grupo 2

## Ejercicio 1:

En este ejercicio se nos pide crear un árbol que tiene dos hijos del proceso padre y otros 3 hijos del segundo hijo del proceso padre.

Hemos usamos varios enteros o arrays de enteros para guardar los pid de cada proceso. Cada vez que creamos un nuevo proceso tenemos que mostrar por pantalla el pid que le corresponde a ese proceso y el pid de sus ancestros. No debemos salir fuera del hijo B así que los mataremos usando kill().

Si la señal la recibe el proceso A o B se ejecutará el comando "pstree", Si la señal la recibe el proceso X o Y se ejecutará el comando "ls".

Además, necesitamos una alarma que espera unos segundos que el usuario pasará por argumento. Finalmente debemos mostrar por pantalla cada pid de los procesos y matarlos, poco a poco de manera que sea de la manera correcta.

A continuación, le mostramos la salida del programa.

Este es el árbol que se crea ejecutado con el comando pstree.



```
Javier@Javier-VirtualBox: ~/Escritorio
Archivo Editar Ver Buscar Terminal Ayuda
javier@javier-VirtualBox:~/Escritorio$ pstree -c | grep ejec
|
|   | -gnome-terminal--+-bash--ejec--ejec--ejec--ejec
|   |   |   |   |
|   |   |   |   | -ejec
|   |   |   |   | -ejec
javier@javier-VirtualBox:~/Escritorio$
```

Este es la salida del programa.

```
javier@javier-VirtualBox: ~/Escritorio
Archivo Editar Ver Buscar Terminal Ayuda
javier@javier-VirtualBox:~$ cd Escritorio/
javier@javier-VirtualBox:~/Escritorio$ ./ejec A 15
Soy el proceso ejec: mi pid es 2670
Soy el proceso A: mi pid es 2671. Mi padre 2670
Soy el proceso B: mi pid es 2672. Mi padre 2671. Mi abuelo 2670
Soy el proceso Z: mi pid es 2675. Mi padre 2672. Mi abuelo 2671. Mi bisabuelo 2670
Soy el proceso Y: mi pid es 2674. Mi padre 2672. Mi abuelo 2671. Mi bisabuelo 2670
Soy el proceso X: mi pid es 2673. Mi padre 2672. Mi abuelo 2671. Mi bisabuelo 2670
/* Tras unos segundos aparecerá */
systemd--ModemManager--({ModemManager})
--({ModemManager})
--NetworkManager--dhclient
--({NetworkManager})
--({NetworkManager})
--VBoxClient--VBoxClient
--VBoxClient--VBoxClient--({VBoxClient})
--VBoxClient--VBoxClient
--VBoxClient--VBoxClient--({VBoxClient})
--VBoxClient--VBoxClient--({VBoxClient})
--VBoxClient--VBoxClient--({VBoxClient})
--VBoxService--({VBoxService})
--({VBoxService})
--({VBoxService})
--({VBoxService})
--({VBoxService})
--({VBoxService})
--accounts-daemon--({accounts-daemon})
--({accounts-daemon})
--acpid
--avahi-daemon--avahi-daemon
--({avahi-daemon})
```

```
javier@javier-VirtualBox: ~/Escritorio
Archivo Editar Ver Buscar Terminal Ayuda
--({gvfsd})
--({gvfsd})
--gvfsd-fuse--({gvfsd-fuse})
--({gvfsd-fuse})
--({gvfsd-fuse})
--({gvfsd-fuse})
--({gvfsd-fuse})
--gvfsd-metadata--({gvfsd-metadata})
--({gvfsd-metadata})
--ibus-portal--({ibus-portal})
--({ibus-portal})
--xdg-permission--({xdg-permission-})
--({xdg-permission-})
--systemd-journal
--systemd-logind
--systemd-resolve
--systemd-udev
--udisksd--({udisksd})
--({udisksd})
--({udisksd})
--({udisksd})
--unattended-upgr--({unattended-upgr})
--upowerd--({upowerd})
--({upowerd})
--whoopsie--({whoopsie})
--({whoopsie})
--wpa_supplicant
Soy Z (2675) y muero
Soy Y (2674) y muero
Soy X (2673) y muero
Soy B (2672) y muero
Soy A (2671) y muero
Soy ejec (2670) y muero
javier@javier-VirtualBox:~/Escritorio$
```

## Ejercicio 2:

Nos pasan dos parámetros el archivo "entrada" y el "salida", donde copiara lo de la "entrada" al "salida".

Para ello creamos una tubería con el comando "pipe", a la hora de abrirla usamos el comando "creat", y le metemos el permiso "0666".

En el bucle lo hace todo de uno en uno debido a que el buffer es de 1.

A continuación, le adjuntamos un foto del código:

```
int main(int argc, char *argv[]) {
    int tuberia[2];
    int archivo;
    char buffer[1];
    pipe(tuberia); //Creamos la tubería

    if(fork() > 0) { //Hijo
        archivo = open(argv[1],O_RDONLY); //Lee el archivo de argv[1](Read only)
        while(read(archivo,buffer,1) > 0) { //Lee archivo
            write(tuberia[1],buffer,1); //Escribe en la tubería
        }
    }
    else{
        archivo = creat(argv[2],0666); //Permiso al programa para: Lectura, Escritura, pero no puede ejecutar
        while(read(tuberia[0],buffer,1) > 0) { //Lee la tubería
            write(archivo, buffer, 1); //Escribe nuevo archivo, copia del original
        }
    }
}
```

## Ejercicio 3:

Lo primero que se nos pasa como parámetro son el número de procesos verticales, seguido del número de procesos horizontales. Con ello creamos el árbol. Después se muestra el primer proceso y los últimos procesos. Por último, se mostrará los padres de los últimos procesos.

A continuación, se muestra un ejemplo de la salida obtenida con los parámetros <hijos 5 3>

```
alu@VDI-Ubuntu-EPS-2017:~/Escritorio$ hijos 5 3
Soy el super padre(4758): mis hijos finales son: 4764 4765 4766
Soy el subhijo 4764, mis padres son: 4760 4759 4761 4762 4763
Soy el subhijo 4765, mis padres son: 4760 4759 4761 4762 4763
Soy el subhijo 4766, mis padres son: 4760 4759 4761 4762 4763
```