# Práctica 1 Sistemas Operativos

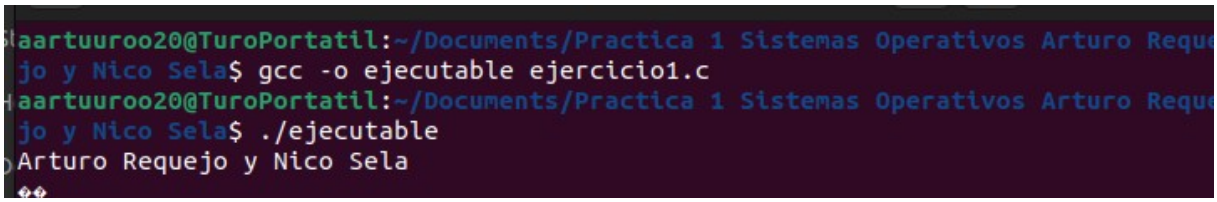
Nicolás Sela Ramos
Arturo Requejo Portilla

UNIVERSIDAD NEBRIJA

# IV. Entregables

## Todos los códigos están en el .zip

### A. Ejercicio 1




Se crean la variables enteras fd y byte_escritos, donde en fd se almacenará la apertura del fichero, y bytes_escritos almacenará la cantidad de bytes que ocupa el archivo. Se crea además una variable de tipo char que servirá como buffer para almacenar los cambios hasta el guardado del archivo. Segundo se establecen los permisos de escritura/ lectura.

Si el archivo no está creado, fd es -1, sino, el fichero está siendo utilizado, no se dispone de los permisos o ya está creado.

Por último, se toma lo escrito en el buffer y se escribe el contenido del buffer en el archivo. Y bytes_escritos almacena el tamaño de este.

### B. Ejercicio 2

Tenemos que realizar dos procesos donde el proceso 1 calcule el cuadrado de los 20 primeros numeros impares mientras que el proceso 2 calcule el cuadrado de los pares. Para ello, hemos realizado un switch, donde va a entrar el proceso 1, en el caso 1 nos da un error debido a que no ha creado el hilo (pid==-1) en el caso 0 donde el pid == 0 se nos crea correctamente el hilo, en este caso, el hilo 1 nos calculará el cuadrado de los numeros impares en el caso por default, donde estamos en el proceso padre creamos el 2do hilo, que nos calculará el cuadrado de los pares. Para ello, a través de un if, comprobamos si se ha creado correctamente, si no se ha creado, da error, y en caso contario, calcula el cuadrado de los pares.

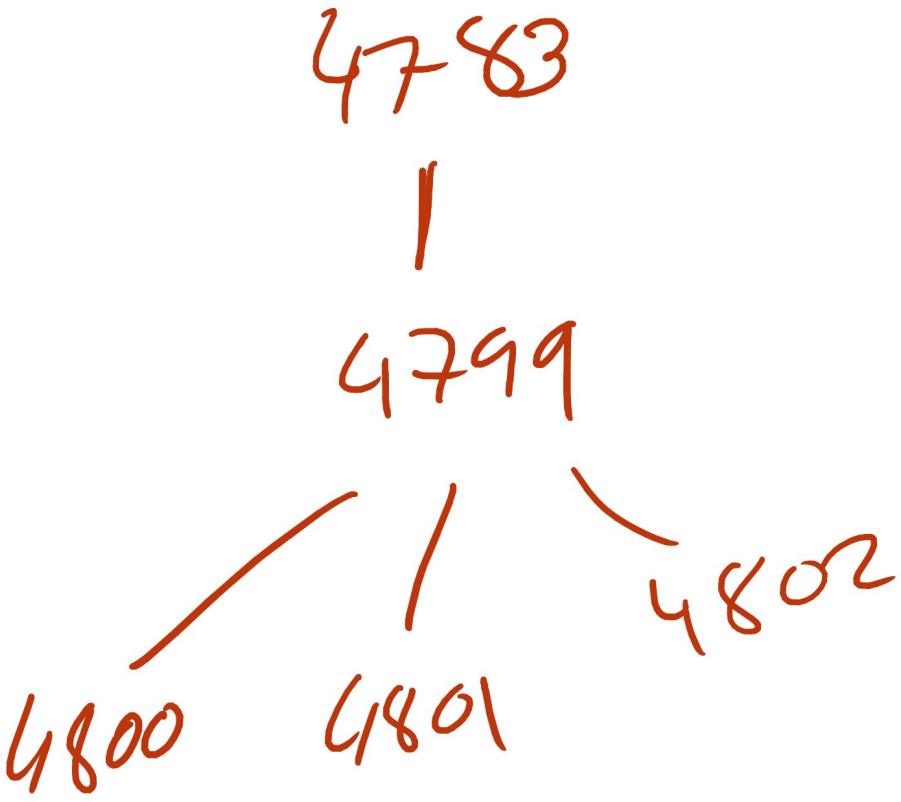
### C. Ejercicio 3

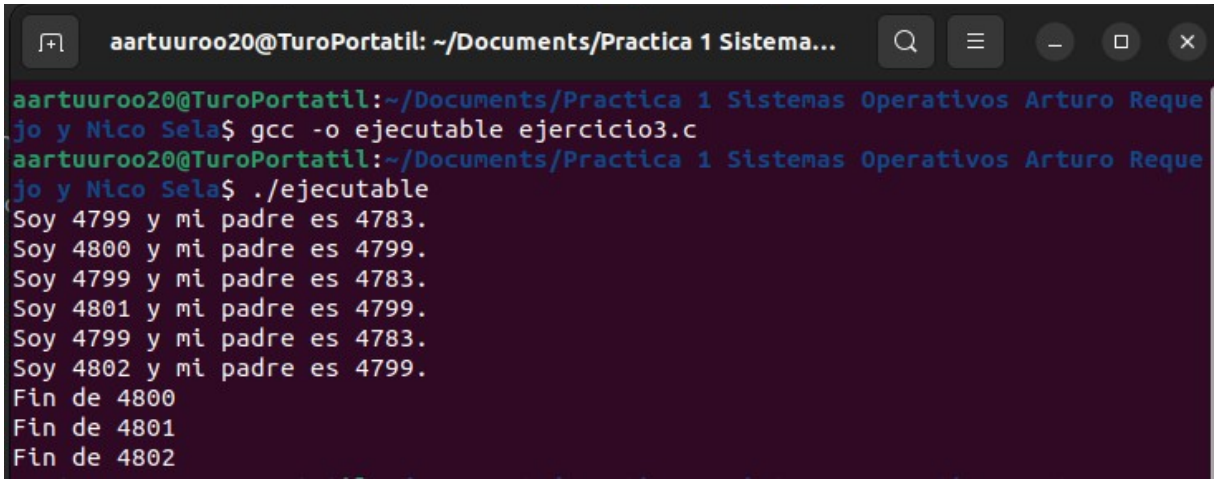
Se generan 3 hilos secundarios o hijos del hilo principal. En la primera linea se puede observar como el proceso kernel (abuelo) tiene un hijo(padre) que es creado

automáticamente por el propio programa. Los 3 hilos hijos son los generados ya por el propio codigo proporcionado en la practica.

Para generar estos procesos hijos generamos id de procesos aleatorios para asignar identificadores al proceso, luego en el propio bucle for estamos generando los 3 hijos aisgnandoles un id de proceso y un padre. Para que no se generen nietos ponemos if(pid == 0) break; para asi cuando estemos en los hijos el bucle se rompa y no continue generando. Este proceso se repite 3 veces. Posteriormente dado que la iteracion 0 (que es el proceso hijo) se ha acabado solo queda parar al padre. Por ende cuando vamos a comprobar si pid == 0 nos quedamos esperando (ya que lo habiamos parado antes) y else  hacemos un bucle que se repita 3 veces y pare los procesos hijos.

En la última línea de código tenemos una instrucción (wait) la cual bloquea al padre hasta que alguno de los hijos haya terminado.

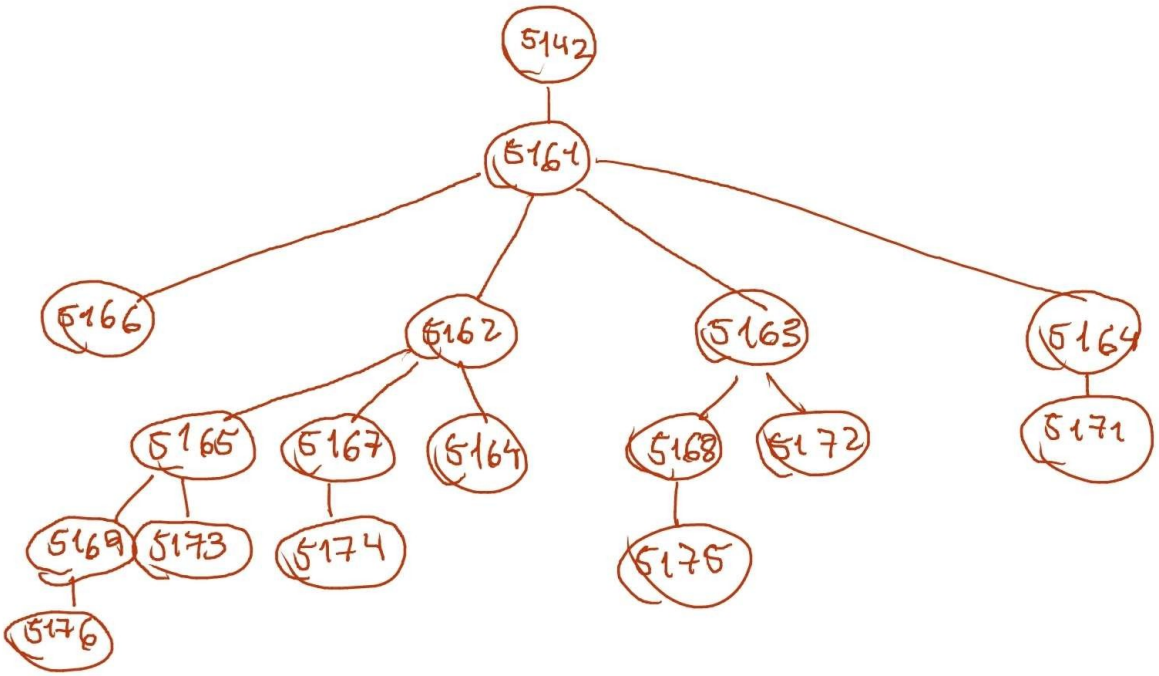





## D. Ejercicio 4

Para el ejercicio 4 sigue el mismo concepto que el ejercicio 3

la diferencia principal es que tenemos que crear 4 hijos del padre y a su vez que estos hijos generen nietos. Para ello tenemos que modificar el ejercicio 3 haciendo que el bucle for se repita 4 veces en vez de 3 para asi se nos generen 4 hijos en vez de 3 y luego quitar el if(pid == 0) break; con esto conseguimos que se nos generen nietos ya que esta instruccion bloquea la creacion de nietos.



```
aartuuroo20@TuroPortatil:~/Documents/Practica 1 Sistemas Operativos Arturo Requejo y Nico Sela$ gcc -o ejecutable ejercicio4.c
aartuuroo20@TuroPortatil:~/Documents/Practica 1 Sistemas Operativos Arturo Requejo y Nico Sela$ ./ejecutable
Soy 5161 y mi padre es 5142.
Soy 5162 y mi padre es 5161.
Soy 5161 y mi padre es 5142.
Soy 5161 y mi padre es 5142.
Soy 5162 y mi padre es 5161.
Soy 5163 y mi padre es 5161.
Soy 5165 y mi padre es 5162.
Soy 5164 y mi padre es 5161.
Soy 5161 y mi padre es 5142.
Soy 5162 y mi padre es 5161.
Soy 5163 y mi padre es 5161.
Soy 5166 y mi padre es 5161.
Soy 5167 y mi padre es 5162.
Soy 5165 y mi padre es 5162.
Soy 5162 y mi padre es 5161.
Soy 5168 y mi padre es 5163.
Soy 5164 y mi padre es 5161.
Soy 5169 y mi padre es 5165.
Soy 5170 y mi padre es 5162.
Soy 5165 y mi padre es 5162.
Soy 5163 y mi padre es 5161.
Soy 5171 y mi padre es 5164.
Soy 5167 y mi padre es 5162.
Soy 5172 y mi padre es 5163.
Soy 5173 y mi padre es 5165.
Soy 5174 y mi padre es 5167.
Soy 5168 y mi padre es 5163.
Soy 5169 y mi padre es 5165.
Soy 5175 y mi padre es 5168.
Soy 5176 y mi padre es 5169.
Fin de 5166
Fin de 5170
Fin de 5173
Fin de 5171
Fin de -1
Fin de -1
Fin de -1
Fin de 5175
Fin de -1
Fin de -1
Fin de -1
Fin de 5172
Fin de 5174
Fin de -1
Fin de -1
Fin de -1
Fin de 5176
Fin de -1
Fin de -1
Fin de -1
Fin de 5164
Fin de 5168
Fin de -1
Fin de -1
Fin de 5167
Fin de 5169
Fin de -1
Fin de -1
Fin de 5163
Fin de 5165
Fin de -1
Fin de 5162
```