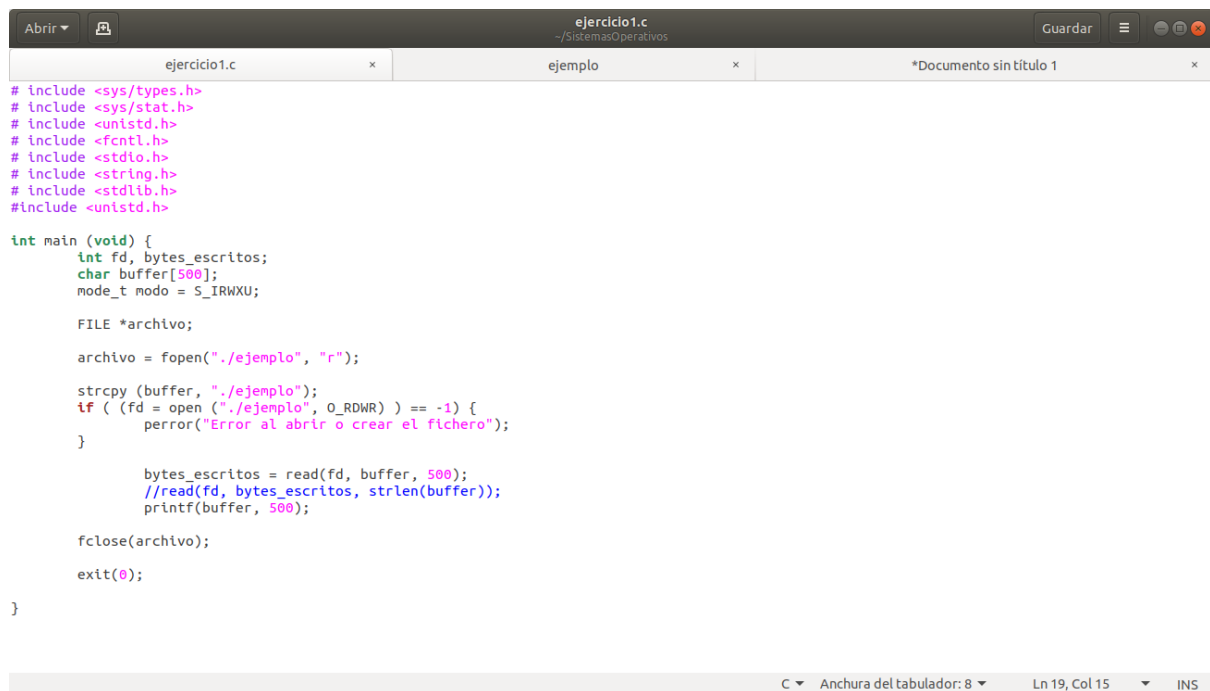


Sistemas Operativos Practica 1

Diego Gonzalez y Carlos Bermúdez

Ejercicio 1:

Llamada al sistema para la apertura y lectura del fichero “ejemplo” en el que deberán aparecer los nombres de los alumnos. Para ello, podrán hacer uso del primer programa mostrado. Justificar el proceso 2 PUNTOS.



```
#include <sys/types.h>
#include <sys/stat.h>
#include <unistd.h>
#include <fcntl.h>
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <unistd.h>

int main (void) {
    int fd, bytes_escritos;
    char buffer[500];
    mode_t modo = S_IRWXU;

    FILE *archivo;

    archivo = fopen("./ejemplo", "r");

    strcpy (buffer, "./ejemplo");
    if ( (fd = open ("../ejemplo", O_RDWR) ) == -1) {
        perror("Error al abrir o crear el fichero");
    }

    bytes_escritos = read(fd, buffer, 500);
    //read(fd, bytes_escritos, strlen(buffer));
    printf(buffer, 500);

    fclose(archivo);

    exit(0);
}
```

El funcionamiento del programa es el siguiente, estamos primero copiando la información del archivo ejemplo en la variable buffer. Después si el fichero no existe se mete en el if y nos sale un error. Después de si se mete en el if o pasa de él mismo nos imprime por pantalla la información que se ha almacenado en el buffer, que en este caso serían los nombres de Diego Gonzalez y Carlos Bermudez. Para terminar el programa debemos cerrar el fichero por posibles errores

Ejercicio 2:

```
alumnos@sistred21-ThinkStation-P320: ~/SistemasOperativos/Ejercicio2
Archivo Editar Ver Buscar Terminal Ayuda
ejercicio2.c: In function 'main':
ejercicio2.c:49:25: warning: implicit declaration of function 'wait'; did you mean 'main'? [-Wimplicit-function-declaration]
    if((childdead = wait(0))!=-1){
                        ^~~~~
                        main
alumnos@sistred21-ThinkStation-P320:~/SistemasOperativos/Ejercicio2$ ./ejecutable
soy el padre ,PID 12536, y voy a esperar a mi hijo(PID 12537)
impares:
1 2 9 4 25 6 49 8 81 10 121 12 169 14 225 16 289 18 361 20

soy el padre ,PID 12537, y voy a esperar a mi hijo(PID 12538)
pares:
1 4 3 16 5 36 7 64 9 100 11 144 13 196 15 256 17 324 19 400

mi hijo con pid 12538, ha muerto
mi hijo con pid 12537, ha muerto
soy el padre ,PID 12536, y voy a esperar a mi hijo(PID 12539)
pares:
1 4 3 16 5 36 7 64 9 100 11 144 13 196 15 256 17 324 19 400

mi hijo con pid 12539, ha muerto
alumnos@sistred21-ThinkStation-P320:~/SistemasOperativos/Ejercicio2$
```

Creamos 2 arrays, uno para pares y otro para impares modificando así solo los números respectivos de cada array, después nos creamos 2 funciones una para realizar el cuadrado de un numero y otra para imprimir el array, luego nos creamos el main, en este, nos creamos 2 pit llamados childpit1 y 2 donde inicializamos con fork cada proceso, saliéndonos así 2 de ellos, en cada proceso nos creamos unos condicionales por si acaso no se crean para gestionar los errores y para ver cuando muere, pero i si se ejecuta, lo que hacemos es recorrerlos el array respectivo para que si el numero contenido él es par o impar dependiendo del proceso sea modificado con su cuadrado para luego imprimirlo, al imprimirlo se muestran los 20 primeros numeros pero modificados solo los requeridos

Ejercicio 3:

```
alumnos@sistred21-ThinkStation-P320: ~/SistemasOperativos/Ejercicio3
Archivo Editar Ver Buscar Terminal Ayuda
alumnos@sistred21-ThinkStation-P320:~/SistemasOperativos/Ejercicio3$ gcc -o ejecutable ejercicio3.c
ejercicio3.c: In function 'main':
ejercicio3.c:26:36: warning: unknown conversion type character '\x0a' in format [-Wformat]
    printf("Fin del proceso de PID%.n", wait(NULL));
                                   ^
ejercicio3.c:26:11: warning: too many arguments for format [-Wformat-extra-args]
    printf("Fin del proceso de PID%.n", wait(NULL));
    ^~~~~~
alumnos@sistred21-ThinkStation-P320:~/SistemasOperativos/Ejercicio3$ ./ejecutable
Soy el proceso de PID 12891 y mi padre tiene 12875 de PID.
Soy el proceso de PID 12891 y mi padre tiene 12875 de PID.
Soy el proceso de PID 12892 y mi padre tiene 12891 de PID.
Soy el proceso de PID 12893 y mi padre tiene 12891 de PID.
Soy el proceso de PID 12891 y mi padre tiene 12875 de PID.
Soy el proceso de PID 12894 y mi padre tiene 12891 de PID.
Fin del proceso de PID%.0
Fin del proceso de PID%.0
Fin del proceso de PID%.0
alumnos@sistred21-ThinkStation-P320:~/SistemasOperativos/Ejercicio3$
```

Aquí lo hacemos es primero tenemos como la raíz el número 12875. Después iniciamos un proceso que es el 12891, que es el proceso 0. Después el siguiente proceso es el 12891 que se llama igual pero es el proceso 1 del padre, después el 12891 tiene como hijo a 12892 que es el proceso 2, a continuación tenemos el proceso 3 que es el 12893, y aquí termina el proceso 1. Ahora continuamos creando el proceso 2 que es el 12891, y en el proceso 2 tiene como hijo 12894 y se crea el proceso 3.