

UNIDAD 1. PROGRAMACIÓN MULTIPROCESO

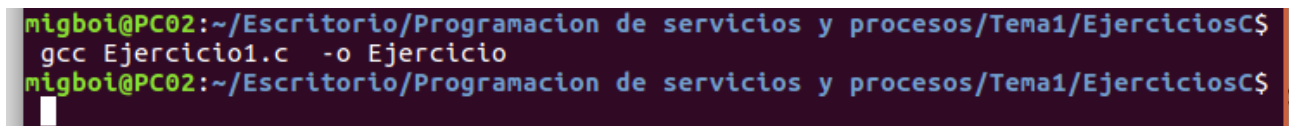
EJERCICIOS GUIADOS DE LA UNIDAD

EJERCICIO 1.

```
#include <stdio.h>
#include <stdlib.h>

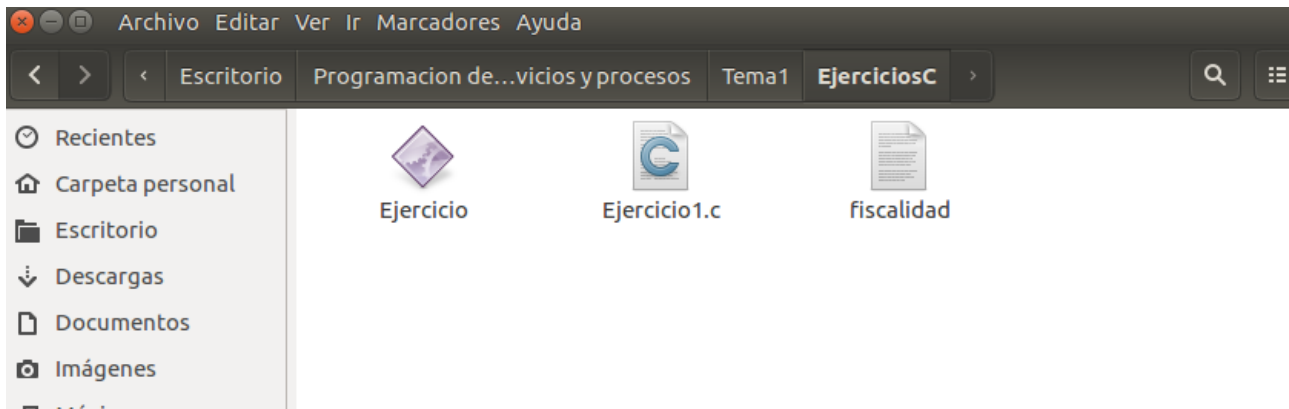
void main(){
    printf("Ejemplo de uso de system():");
    printf("\n\tListado del directorio actual y envio del fichero:");
    printf("%d",system("ls > fiscalidad"));
    printf("\n\tAbrimos con el gedit el fichero ");

    printf("%d",system("gedit ficsalida"));
    printf("\n\tEste Comado es erroneo: %d",system("ged"));
    printf("\nFin de programa...\n");
}
```

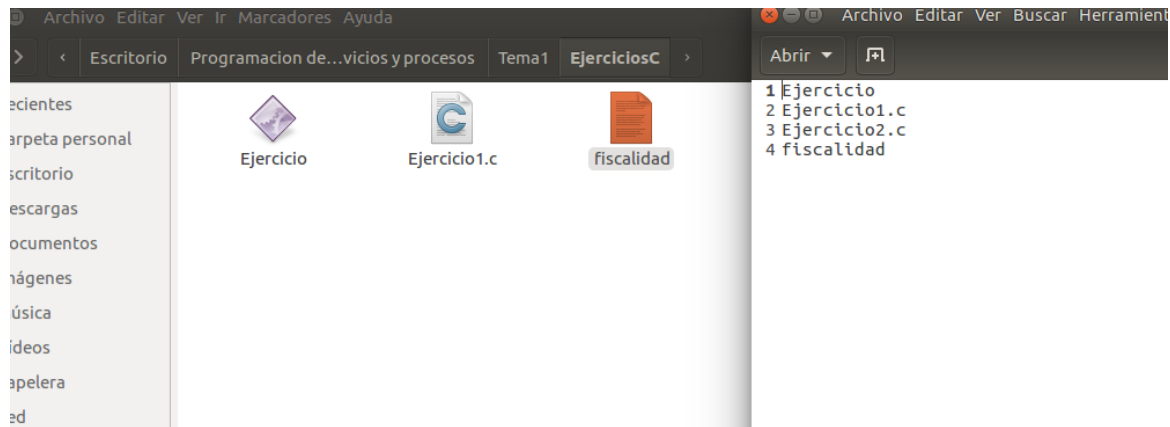


```
migbol@PC02:~/Escritorio/Programacion de servicios y procesos/Tema1/EjerciciosC$
gcc Ejercicio1.c -o Ejercicio
migbol@PC02:~/Escritorio/Programacion de servicios y procesos/Tema1/EjerciciosC$
```

System() , Sirve para ejecutar suberosos o **comandos** del sistema operativo.



Después con ls y una tubería hacia el fichero fiscalizad listamos y guardamos todos los fichero dentro del directorio.



EJERCICIO 2

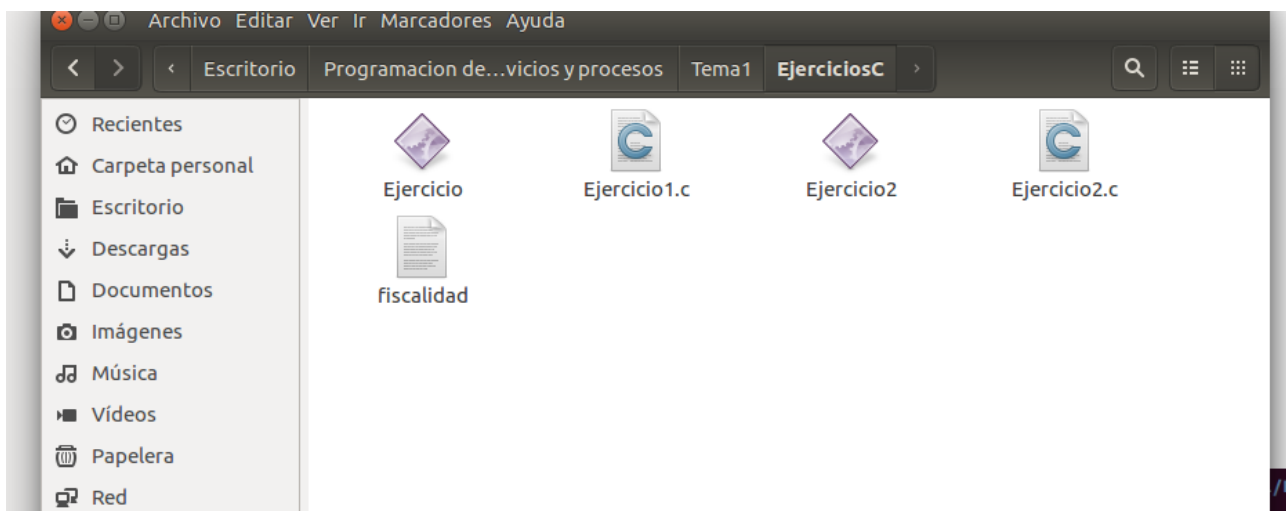
```
#include <unistd.h>
```

```
#include <stdio.h>
```

```
void main(){
    printf("Ejemplo de uso exec():");
    printf("Los archivos en el directorio son: \n");
    execl("/bin/ls", "ls", "-l", (char*)NULL);
    printf("!!! Esto no se ejecuta !!! \n")
}
```

```
gcc Ejercicio2.c -o Ejercicio2
migboi@PC02:~/Escritorio/Programacion de servicios y procesos/Tema1/EjerciciosC$
gcc Ejercicio2.c -o Ejercicio2
migboi@PC02:~/Escritorio/Programacion de servicios y procesos/Tema1/EjerciciosC$
```

Se crea un archivo ejecutable



Muestra todo lo que hay en tu directorio con los permisos y quien lo ha creado.

```
migboi@PC02:~/Escritorio/Programacion de servicios y procesos/Tema1/EjerciciosC$ ./Ejercicio2
Ejemplo de uso exec(): Los archivos en el directorio son:
total 56
-rw-r--r-- 1 migboi students 709 sep 19 09:06 ejemploFork.c
-rw-r--r-- 1 migboi students 261 sep 19 08:44 ejemploPadres.c
-rwxr-xr-x 1 migboi students 8712 sep 19 08:17 Ejercicio
-rw-r--r-- 1 migboi students 416 sep 19 08:21 Ejercicio1.c
-rwxr-xr-x 1 migboi students 8712 sep 19 08:32 Ejercicio2
-rw-r--r-- 1 migboi students 238 sep 19 08:32 Ejercicio2.c
-rwxr-xr-x 1 migboi students 8720 sep 19 08:46 ejemploPadres
-rw-r--r-- 1 migboi students 87 sep 19 08:41 fiscalidad
migboi@PC02:~/Escritorio/Programacion de servicios y procesos/Tema1/EjerciciosC$
```

EJERCICIO 3

```
#include <stdio.h>
#include <unistd.h>

void main(void)
{
    pid_t id_pactual, id_padre;

    id_pactual= getpid();
    id_padre= getppid();

    printf("PID de este procesos: %d\n",id_pactual);
    printf("PID del procesos padre: %d\n ",id_pactual);
}
```

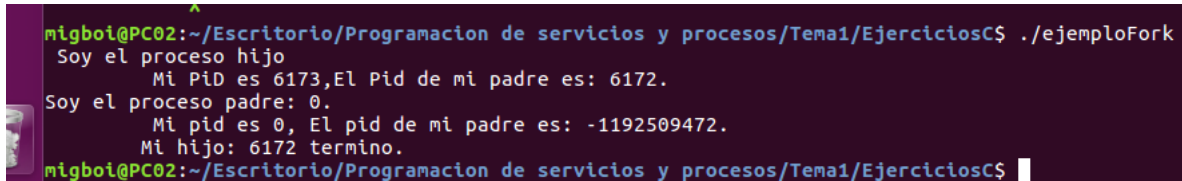
Lo compilamos con : gcc ejemploPadres.c -o ejmploPadres

despues lo ejecutamos con: ./ejmploPadres

```
migboi@PC02:~/Escritorio/Programacion de servicios y procesos/Tema1/EjerciciosC$
./ejmploPadres
PID de este procesos: 5573
PID del procesos padre: 5573
migboi@PC02:~/Escritorio/Programacion de servicios y procesos/Tema1/EjerciciosC$
```

EJERCICIO 4.

El comando `fork()` crea un proceso hijo idéntico a el.



```
migboi@PC02:~/Escritorio/Programacion de servicios y procesos/Tema1/EjerciciosCS$ ./ejemploFork
Soy el proceso hijo
  Mi PiD es 6173,El Pid de mi padre es: 6172.
Soy el proceso padre: 0.
  Mi pid es 0, El pid de mi padre es: -1192509472.
  Mi hijo: 6172 termino.
migboi@PC02:~/Escritorio/Programacion de servicios y procesos/Tema1/EjerciciosCS$
```

```
#include <stdlib.h>
#include <unistd.h>
#include <stdio.h>
```

```
void main(){
```

```
    pid_t pid, Hijo_pid;
    pid = fork();
```

```
    if(pid == -1)//Ha ocurrido un error
    {
        printf("No se ha podido crear el proceso hijo ...");
        exit(-1);
    }
```

```
    if(pid == 0) //Nos encontramos en Proceso hijo
    {
        printf(" Soy el proceso hijo \n\t");
        printf(" Mi PiD es %d,El Pid de mi padre es: %d.\n",getpid(),getppid());
    }
```

```
    else //Nos encontramos en proceso padre
    {
        Hijo_pid = wait(NULL); //espera la finalizacion del proceso hijo
        printf("Soy el proceso padre: %d.\n\t Mi pid es %d, El pid de mi padre es: %d.\n\t");
        printf("Mi hijo: %d termino. \n",getpid(),getppid(),pid );
    }
```

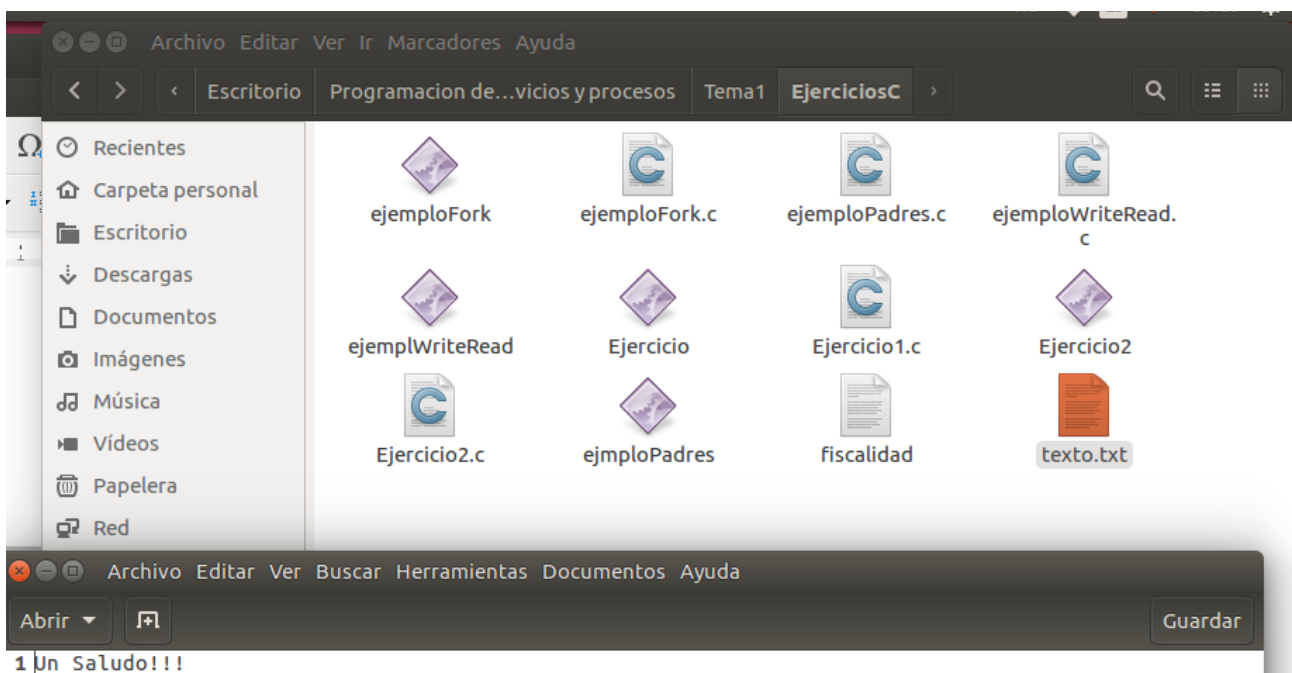
```
    exit(0);
```

```
}
```

EJERCICIO 5

```
migboi@PC02:~/Escritorio/Programacion de servicios y procesos/Tema1/EjerciciosC$  
./ejemplWriteRead  
Escribiendo el saludo ...  
Contenido del Fichero:  
Violación de segmento ('core' generado)  
migboi@PC02:~/Escritorio/Programacion de servicios y procesos/Tema1/EjerciciosC$
```

Hay que crear primero el txt en la misma carpeta que este el ejecutable .
Este programa l guarda en el saludo en fichero y lo lee.



Ejercicio 9

```
#include <stdio.h>
#include <stdlib.h>
```

```
void main(){
```

```
printf("%d",system("ps -aux > ejecucion9"));
```

```
printf("%d",system("nano.ejecucion9"));
```

```
};
```

ejercicio9.c						ejecucion9			
PID	%CPU	%MEM	VSZ	RSS	TTY	STAT	START	TIME	COMMAND
1	0.0	0.0	185356	5944	?	Ss	08:02	0:01	/sbin/init splash
2	0.0	0.0	0	0	?	S	08:02	0:00	[kthreadd]
4	0.0	0.0	0	0	?	I<	08:02	0:00	[kworker/0:0H]
6	0.0	0.0	0	0	?	I<	08:02	0:00	[mm_percpu_wq]
7	0.0	0.0	0	0	?	S	08:02	0:00	[ksoftirqd/0]
8	0.0	0.0	0	0	?	I	08:02	0:08	[rcu_sched]
9	0.0	0.0	0	0	?	I	08:02	0:00	[rcu_bh]
10	0.0	0.0	0	0	?	S	08:02	0:00	[migration/0]
11	0.0	0.0	0	0	?	S	08:02	0:00	[watchdog/0]
12	0.0	0.0	0	0	?	S	08:02	0:00	[cpuhp/0]
13	0.0	0.0	0	0	?	S	08:02	0:00	[cpuhp/1]
14	0.0	0.0	0	0	?	S	08:02	0:00	[watchdog/1]
15	0.0	0.0	0	0	?	S	08:02	0:00	[migration/1]
16	0.0	0.0	0	0	?	S	08:02	0:00	[ksoftirqd/1]
18	0.0	0.0	0	0	?	I<	08:02	0:00	[kworker/1:0H]
19	0.0	0.0	0	0	?	S	08:02	0:00	[cpuhp/2]
20	0.0	0.0	0	0	?	S	08:02	0:00	[watchdog/2]
21	0.0	0.0	0	0	?	S	08:02	0:00	[migration/2]
22	0.0	0.0	0	0	?	S	08:02	0:00	[ksoftirqd/2]
24	0.0	0.0	0	0	?	I<	08:02	0:00	[kworker/2:0H]
25	0.0	0.0	0	0	?	S	08:02	0:00	[cpuhp/3]
26	0.0	0.0	0	0	?	S	08:02	0:00	[watchdog/3]
27	0.0	0.0	0	0	?	S	08:02	0:00	[migration/3]
28	0.0	0.0	0	0	?	S	08:02	0:00	[ksoftirqd/3]
30	0.0	0.0	0	0	?	I<	08:02	0:00	[kworker/3:0H]
31	0.0	0.0	0	0	?	S	08:02	0:00	[kdevtmpfs]
32	0.0	0.0	0	0	?	I<	08:02	0:00	[netns]
33	0.0	0.0	0	0	?	S	08:02	0:00	[rcu_tasks_kthre]
34	0.0	0.0	0	0	?	S	08:02	0:00	[kauditd]
38	0.0	0.0	0	0	?	S	08:02	0:00	[khungtaskd]
39	0.0	0.0	0	0	?	S	08:02	0:00	[oom_reaper]
40	0.0	0.0	0	0	?	I<	08:02	0:00	[writeback]
41	0.0	0.0	0	0	?	S	08:02	0:00	[kcompactd0]

Texto plano ▾ Anchura de la pestaña: 4 ▾ Ln 1, Col 1 ▾

mentos

Ejercicio10.c

nes

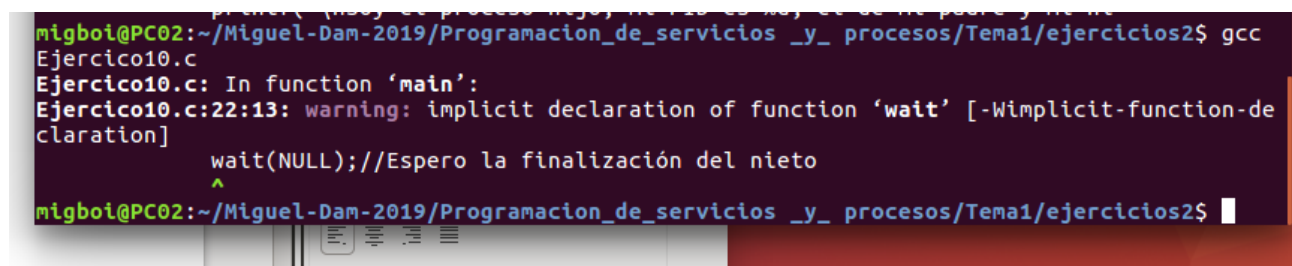
Ejercicio10

```
#include <stdlib.h>
#include <unistd.h>
#include <stdio.h>

void main(){
    pid_t pid, hijo_pid, nieto_pid;
    pid = fork();

    if(pid == -1){
        printf("No se ha podido crear el proceso hijo...");
        exit(-1);
    }
    if(pid == 0){
        nieto_pid = fork();
        if(nieto_pid == -1){
            printf("Error creando el proceso nieto");
            exit(-1);
        }
        if(nieto_pid == 0){
            printf("\nSoy el proceso nieto, mi PID es %d, el PID de mi padre es %d", getpid(),
getppid());
        }else{
            wait(NULL); //Espero la finalización del nieto
            printf("\nSoy el proceso hijo, mi PID es %d, el de mi padre %d y mi hijo(el nieto) es %d",
getpid(),getppid(),nieto_pid);
        }
    }else{
        hijo_pid = wait(NULL); //Espera la finalización del proceso hijo
        printf("\nSoy el proceso padre: \n\t Mi PID es %d, el PID de mi padre es: %d.\n\t Mi hijo %d
terminó.\n",getpid(), getppid(), pid);
    }
    exit(0);
}
```

no me funciona



```
migboi@PC02:~/Miguel-Dam-2019/Programacion_de_servicios_y_procesos/Tema1/ejercicios2$ gcc
Ejercicio10.c
Ejercicio10.c: In function 'main':
Ejercicio10.c:22:13: warning: implicit declaration of function 'wait' [-Wimplicit-function-de
claration]
    wait(NULL); //Espero la finalización del nieto
    ^
migboi@PC02:~/Miguel-Dam-2019/Programacion_de_servicios_y_procesos/Tema1/ejercicios2$
```

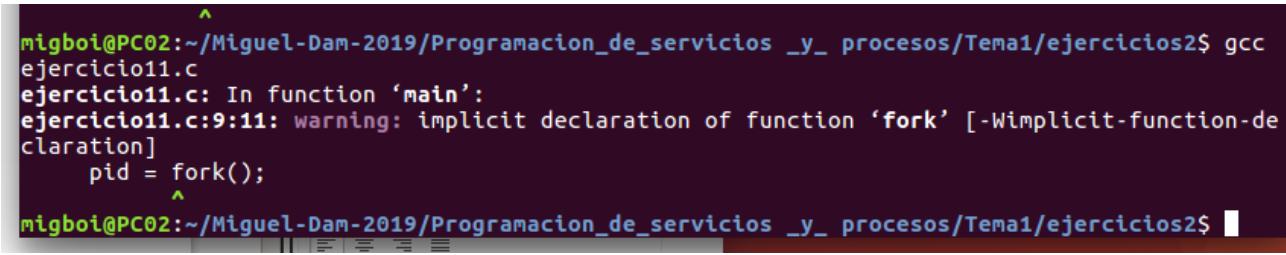
Ejercicio 11

```
#include <stdlib.h>
#include <stdio.h>

void main(void){
    pid_t pid;
    int x = 6;

    printf("Valor inicial de la variable: %d\n", x);
    pid = fork();

    switch(pid) {
        case -1: //ERROR
            printf("NO SE HA PODIDO CREAR HIJO...");
            exit(-1);
        case 0: //HIJO
            x = x - 5;
            printf("Variable en proceso hijo: %d\n", x);
            break;
        default: //PADRE
            x = x + 5;
            printf("Variable en proceso padre: %d\n", x);
            break;
    }
}
```



```
migboi@PC02:~/Miguel-Dam-2019/Programacion_de_servicios_y_procesos/Tema1/ejercicios2$ gcc
ejercicio11.c
ejercicio11.c: In function 'main':
ejercicio11.c:9:11: warning: implicit declaration of function 'fork' [-Wimplicit-function-de
claration]
    pid = fork();
           ^
migboi@PC02:~/Miguel-Dam-2019/Programacion_de_servicios_y_procesos/Tema1/ejercicios2$
```