

Modulo-II-Sesion-5.pdf



KIKONASO



Sistemas Operativos



2º Grado en Ingeniería Informática



**Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación
Universidad de Granada**



Escuela de
organización
Industrial

MÁSTER EN

**Inteligencia Artificial
& Data Management**

MADRID

Formamos
talento para un futuro
Sostenible

saber más



Esto no son apuntes pero **tiene un 10 asegurado** (y lo vas a disfrutar igual).

Abre la Cuenta NoCuenta con el código **WUOLAH10**, haz tu primer pago y llévate 10 €.

Me interesa



1/6

Este número es indicativo del riesgo del producto, siendo 1/6 indicativo de menor riesgo y 6/6 de mayor riesgo.

ING BANK NV se encuentra adherido al Sistema de Garantía de Depósitos Holandeses con una garantía de hasta 100.000 euros por depositante. Consulta más información en [ing.es](https://www.ing.es)

Módulo II. Uso de los Servicios del SO mediante la API

Sesión 5. Llamadas al sistema para gestión y control de señales

Ejercicio 1. Compila y ejecuta los siguientes programas y trata de entender su funcionamiento.

El programa `envioSignal.c`, como bien dice su nombre sirve para enviar señales a un proceso activo con un PID específico, en el que se si ejecutamos:

- `./envio 0 <PID>` : Enviará la señal `SIGTERM` a dicho proceso, lo que hará que finalice.
- `./envio 1 <PID>` : Enviará la señal `SIGUSR1` a dicho proceso, la cual tendrá que ser definida en un manejador por el usuario para que haga algo en específico.
- `./envio 2 <PID>` : Enviará la señal `SIGUSR2` a dicho proceso, la cual también tendrá que ser definida en un manejador por el usuario para que haga algo en específico.

En el programa `reciboSignal.c` lo que se hace es crear un manejador para que gestione el comportamiento de un proceso cuando se reciba una señal `SIGUSR1` o `SIGUSR2`, siendo los casos:

- `SIGUSR1` : Imprime el mensaje `"\nRecibida la señal SIGUSR1\n\n"`
- `SIGUSR2` : Imprime el mensaje `"\nRecibida la señal SIGUSR2\n\n"`

El bucle `for` del final solo se encuentra ahí para que el proceso no termine y pueda estar gestionando las señales mencionadas indefinidamente.

Ahora, para probar la ejecución de los programas, haremos:

Compilamos

```
gcc envioSignal.c -o envio
gcc reciboSignal.c -o recibo
```

Ejecuto `reciboSignal` en una terminal para obtener su PID:

```
./recibo
```

Luego, verifico su PID usando:

```
ps -aux | grep recibo
```

Y ahora envío señales desde otra terminal:

```
./envio 1 <PID> # Envía SIGUSR1
./envio 2 <PID> # Envía SIGUSR2
./envio 0 <PID> # Envía SIGTERM
```

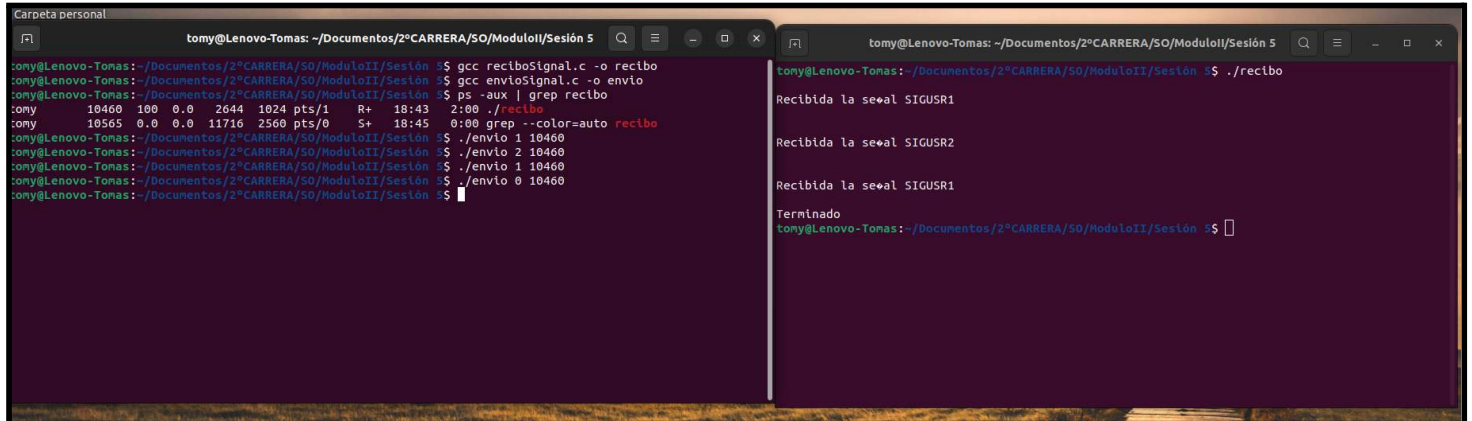
Consulta
condiciones aquí



do your thing

WUOLAH

Ya solo queda mostrar los resultados:



```
tomy@Lenovo-Tomas: ~/Documentos/2°CARRERA/SO/ModuloII/Sesión 5
tomy@Lenovo-Tomas:~/Documentos/2°CARRERA/SO/ModuloII/Sesión 5$ gcc reciboSignal.c -o recibo
tomy@Lenovo-Tomas:~/Documentos/2°CARRERA/SO/ModuloII/Sesión 5$ gcc envioSignal.c -o envio
tomy@Lenovo-Tomas:~/Documentos/2°CARRERA/SO/ModuloII/Sesión 5$ ps -aux | grep recibo
tomy 10460 100 0.0 2644 1024 pts/1 R+ 18:43 2:00 ./recibo
tomy 10565 0.0 0.0 11716 2560 pts/0 S+ 18:45 0:00 grep --color=auto recibo

tomy@Lenovo-Tomas:~/Documentos/2°CARRERA/SO/ModuloII/Sesión 5$ ./envio 1 10460
tomy@Lenovo-Tomas:~/Documentos/2°CARRERA/SO/ModuloII/Sesión 5$ ./envio 2 10460
tomy@Lenovo-Tomas:~/Documentos/2°CARRERA/SO/ModuloII/Sesión 5$ ./envio 1 10460
tomy@Lenovo-Tomas:~/Documentos/2°CARRERA/SO/ModuloII/Sesión 5$ ./envio 0 10460
tomy@Lenovo-Tomas:~/Documentos/2°CARRERA/SO/ModuloII/Sesión 5$

tomy@Lenovo-Tomas:~/Documentos/2°CARRERA/SO/ModuloII/Sesión 5$ ./recibo
Recibida la se al SIGUSR1

Recibida la se al SIGUSR2

Recibida la se al SIGUSR1

Terminado
tomy@Lenovo-Tomas:~/Documentos/2°CARRERA/SO/ModuloII/Sesión 5$
```

Ejercicio 2. Escribe un programa en C llamado contador, tal que cada vez que reciba una se al que se pueda manejar, muestre por pantalla la se al y el n mero de veces que se ha recibido ese tipo de se al, y un mensaje inicial indicando las se ales que no puede manejar. En el cuadro siguiente se muestra un ejemplo de ejecuci n del programa:

```
kawtar@kawtar-VirtualBox:~$ ./contador &

[2] 1899

kawtar@kawtar-VirtualBox:~$

No puedo manejar la se al 9

No puedo manejar la se al 19

Esperando el env o de se ales...

kill -SIGINT 1899

kawtar@kawtar-VirtualBox:~$ La se al 2 se ha recibido 1 veces

kill -SIGINT 1899

La se al 2 se ha recibido 2 veces

kill -15 1899

kawtar@kawtar-VirtualBox:~$ La se al 15 se ha recibido 1 veces

kill -111 1899

bash: kill: 111: especificaci n de se al inv lida

kawtar@kawtar-VirtualBox:~$ kill -15 1899 // el programa no puede capturar la se al 15

[2]+ Detenido ./contador

kawtar@kawtar-VirtualBox:~$ kill -cont 1899

La se al 18 se ha recibido 1 veces

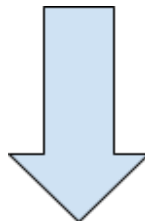
kawtar@kawtar-VirtualBox:~$ kill -KILL 1899
```

```

1  /*
2  contador.c
3  Ejercicio 2. Escribe un programa en C llamado contador, tal que cada vez que reciba una
4  señal que se pueda manejar, muestre por pantalla la señal y el número de veces que se ha
5  recibido ese tipo de señal, y un mensaje inicial indicando las señales que no puede manejar.
6  */
7
8  #include <sys/types.h>
9  #include <unistd.h>
10 #include <stdio.h>
11 #include <signal.h>
12 #include <stdlib.h>
13 #include <errno.h>
14 #include <string.h>
15
16 #define NUM_SENALES_TOTAL 64
17
18 static int contador[NUM_SENALES_TOTAL]={0};
19
20 static void manejador(int sigNum)
21 {
22     if (sigNum != SIGSTOP && sigNum!= SIGKILL){
23         contador[sigNum-1]++;
24         printf(" \n La señal %d ha sido recibida %d veces \n", sigNum, contador[sigNum-1]);
25     }
26 }
27
28 }
29
30 int main(int argc, char *argv[])
31 {
32     struct sigaction sigact;
33
34     if(setvbuf(stdout,NULL,_IONBF,0))
35     {
36         perror("\nError en setvbuf");
37     }
38
39     sigact.sa_handler= manejador;
40     sigemptyset (&sigact.sa_mask);
41     sigact.sa_flags = 0;
42
43     printf("No puedo manejar las señales SIGKILL (%d) y SIGSTOP (%d) \n", SIGKILL, SIGSTOP);
44     printf("Esperando el envío de señales... \n");
45
46     //Configuramos el manejador para todas las señales:
47
48     for(int i=1; i < NUM_SENALES_TOTAL; ++i ){
49         if (i != SIGKILL && i!= SIGSTOP && i!=32 && i!= 33){ // Omitir SIGRTMIN y SIGRTMIN+1
50             if (sigaction(i, &sigact, NULL) < 0) {
51                 if (sigaction(i, &sigact, NULL) < 0 ){
52                     fprintf(stderr, "No se pudo manejar la señal %d: %s\n", i, strerror(errno));
53                 }
54             }
55         }
56     }
57
58     for(;;)
59     {
60     }
61 }
62 }
63

```

Un ejemplo de ejecución del programa sería:



Esto no son apuntes pero tiene un 10 asegurado (y lo vas a disfrutar igual).

Abre la Cuenta NoCuenta con el código **WUOLAH10**, haz tu primer pago y llévate 10 €.

Me interesa

1/6

Este número es indicativo del riesgo del producto, siendo 1/6 indicativo de menor riesgo y 6/6 de mayor riesgo.

ING BANK NV se encuentra adherido al Sistema de Garantía de Depósitos Holandés con una garantía de hasta 100.000 euros por depositante. Consulta más información en [ing.es](https://www.ing.es)



```
tomy@Lenovo-Tomas: ~/Documentos/2ºCARRERA/S0/ModuloII/Sesión 5$ kill -l
1) SIGHUP      2) SIGINT      3) SIGQUIT     4) SIGILL      5) SIGTRAP
6) SIGABRT     7) SIGBUS      8) SIGFPE     9) SIGKILL    10) SIGUSR1
11) SIGSEGV    12) SIGUSR2    13) SIGPIPE   14) SIGALRM    15) SIGTERM
16) SIGSTKFLT  17) SIGCHLD    18) SIGCONT   19) SIGSTOP    20) SIGTSTP
21) SIGTTIN    22) SIGTTOU    23) SIGURG    24) SIGXCPU    25) SIGXFSZ
26) SIGVTALRM  27) SIGPROF   28) SIGWINCH  29) SIGIO      30) SIGPWR
31) SIGSYS     34) SIGRTMIN   35) SIGRTMIN+1 36) SIGRTMIN+2 37) SIGRTMIN+3
38) SIGRTMIN+4 39) SIGRTMIN+5 40) SIGRTMIN+6 41) SIGRTMIN+7 42) SIGRTMIN+8
43) SIGRTMIN+9 44) SIGRTMIN+10 45) SIGRTMIN+11 46) SIGRTMIN+12 47) SIGRTMIN+13
48) SIGRTMIN+14 49) SIGRTMIN+15 50) SIGRTMAX-14 51) SIGRTMAX-13 52) SIGRTMAX-12
53) SIGRTMAX-11 54) SIGRTMAX-10 55) SIGRTMAX-9 56) SIGRTMAX-8 57) SIGRTMAX-7
58) SIGRTMAX-6 59) SIGRTMAX-5 60) SIGRTMAX-4 61) SIGRTMAX-3 62) SIGRTMAX-2
63) SIGRTMAX-1 64) SIGRTMAX

tomy@Lenovo-Tomas: ~/Documentos/2ºCARRERA/S0/ModuloII/Sesión 5$ gcc contador.c -o contador
tomy@Lenovo-Tomas: ~/Documentos/2ºCARRERA/S0/ModuloII/Sesión 5$ ./contador &
[2] 12443
tomy@Lenovo-Tomas: ~/Documentos/2ºCARRERA/S0/ModuloII/Sesión 5$ No puedo manejar las señales SIGKILL
Esperando el envío de señales...
kill -2 12443

La señal 2 ha sido recibida 1 veces
tomy@Lenovo-Tomas: ~/Documentos/2ºCARRERA/S0/ModuloII/Sesión 5$ kill -2 12443

La señal 2 ha sido recibida 2 veces
tomy@Lenovo-Tomas: ~/Documentos/2ºCARRERA/S0/ModuloII/Sesión 5$ kill -2 12443

La señal 2 ha sido recibida 3 veces
tomy@Lenovo-Tomas: ~/Documentos/2ºCARRERA/S0/ModuloII/Sesión 5$ kill -SIGCONT 12443

La señal 18 ha sido recibida 1 veces
tomy@Lenovo-Tomas: ~/Documentos/2ºCARRERA/S0/ModuloII/Sesión 5$ kill -SIGCONT 12443

La señal 18 ha sido recibida 2 veces
tomy@Lenovo-Tomas: ~/Documentos/2ºCARRERA/S0/ModuloII/Sesión 5$ kill -SIGHUP 12443

La señal 1 ha sido recibida 1 veces
tomy@Lenovo-Tomas: ~/Documentos/2ºCARRERA/S0/ModuloII/Sesión 5$ kill -SIGHUP 12443

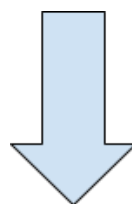
La señal 1 ha sido recibida 2 veces
tomy@Lenovo-Tomas: ~/Documentos/2ºCARRERA/S0/ModuloII/Sesión 5$ kill -SIGTRAP 12443

La señal 5 ha sido recibida 1 veces
tomy@Lenovo-Tomas: ~/Documentos/2ºCARRERA/S0/ModuloII/Sesión 5$ kill -SIGFPE 12443

La señal 8 ha sido recibida 1 veces
tomy@Lenovo-Tomas: ~/Documentos/2ºCARRERA/S0/ModuloII/Sesión 5$ kill -SIGTRAP 12443

La señal 5 ha sido recibida 2 veces
tomy@Lenovo-Tomas: ~/Documentos/2ºCARRERA/S0/ModuloII/Sesión 5$ kill -SIGKILL 12443
tomy@Lenovo-Tomas: ~/Documentos/2ºCARRERA/S0/ModuloII/Sesión 5$ kill -SIGTRAP 12443
bash: kill: (12443) - No existe el proceso
[2]+ Terminado (killed) ./contador
tomy@Lenovo-Tomas: ~/Documentos/2ºCARRERA/S0/ModuloII/Sesión 5$
```

Ejercicio 3. Escribe un programa que suspenda la ejecución del proceso actual hasta que se reciba la señal SIGUSR1. Consulta en el manual en línea [sigemptyset](https://man7.org/linux/man-pages/SIGEMTSET.7.html) para conocer las distintas operaciones que permiten configurar el conjunto de señales de un proceso.



Consulta condiciones aquí



do your thing

WUOLAH

```

1  /*
2  ejercicio3.c
3  Ejercicio 3. Escribe un programa que suspenda la ejecución del proceso actual hasta que se reciba
4  la señal SIGUSR1. Consulta en el manual en línea sigemptyset para conocer las distintas operaciones
5  que permiten configurar el conjunto de señales de un proceso.
6
7  */
8
9  #include <stdio.h>
10 #include <signal.h>
11 #include <unistd.h>
12
13 // Manejador para la señal SIGUSR1
14 void handle_sigusr1(int signo) {
15     printf("Señal SIGUSR1 recibida.\n");
16 }
17
18 int main() {
19     sigset_t new_mask, old_mask;
20
21     // Configurar el manejador para SIGUSR1
22     struct sigaction sa;
23     sa.sa_handler = handle_sigusr1; // Usar el manejador
24     sigemptyset(&sa.sa_mask);       // No bloquear otras señales durante el manejo
25     sa.sa_flags = 0;                 // Sin banderas adicionales
26     sigaction(SIGUSR1, &sa, NULL);
27
28     // Inicializar la nueva máscara de señales
29     sigemptyset(&new_mask);
30     sigaddset(&new_mask, SIGUSR1); // Agregar SIGUSR1 al conjunto de señales a bloquear
31
32     // Bloquear SIGUSR1 temporalmente
33     sigprocmask(SIG_BLOCK, &new_mask, &old_mask);
34
35     printf("Esperando señal SIGUSR1...\n");
36
37     // Usar sigsuspend para esperar SIGUSR1
38     sigset_t suspend_mask;
39     sigemptyset(&suspend_mask); // Conjunto vacío desbloquea todo menos lo bloqueado previamente
40     sigsuspend(&suspend_mask);
41
42     // Restaurar la máscara de señales original
43     sigprocmask(SIG_SETMASK, &old_mask, NULL);
44
45     printf("El proceso se reanudó después de recibir SIGUSR1.\n");
46     return 0;
47 }
48

```

```

kill -SIGUSR1 13816
tomy@Lenovo-Tomas:~/Documentos/2°CARRERA/S0/ModuloII/Sesión 5$ gcc ejercicio3.c -o ej3
tomy@Lenovo-Tomas:~/Documentos/2°CARRERA/S0/ModuloII/Sesión 5$ ./ej3 &
[2] 13816
tomy@Lenovo-Tomas:~/Documentos/2°CARRERA/S0/ModuloII/Sesión 5$ Esperando señal SIGUSR1.
kill -SIGUSR1 13816
Señal SIGUSR1 recibida.
El proceso se reanudó después de recibir SIGUSR1.
tomy@Lenovo-Tomas:~/Documentos/2°CARRERA/S0/ModuloII/Sesión 5$

```

Ejercicio 4. Compila y ejecuta el siguiente programa y trata de entender su funcionamiento.

//tarea12.c

Resumen del programa

1. Manejador de señales:

1/6

Este número es indicativo del riesgo del producto, siendo 1/6 indicativo de menor riesgo y 6/6 de mayor riesgo.

ING BANK NV se encuentra adherido al Sistema de Garantía de Depósitos Holandés con una garantía de hasta 100.000 euros por depositante. Consulta más información en ing.es

Que te den **10 € para gastar**
es una fantasía.
ING lo hace realidad.

Abre la **Cuenta NoCuenta** con el código
WUOLAH10, haz tu primer pago y llévate 10 €.

Quiero el cash

[Consulta condiciones aquí](#)



do your thing

- Se define un manejador (**manejador**) para la señal **SIGTERM**. Este establece una bandera (**signal_recibida**) en 1 cuando se recibe la señal.
- 2. **Bloqueo de señales:**
 - Se bloquea temporalmente **SIGTERM** utilizando **sigprocmask**. Durante este tiempo, cualquier intento de enviar **SIGTERM** al proceso será almacenado en espera (en lugar de ser procesado inmediatamente).
- 3. **Suspensión del proceso:**
 - El programa duerme durante 10 segundos con **sleep(10)**. Durante este tiempo, la señal **SIGTERM** está bloqueada, pero si se envía, se queda pendiente.
- 4. **Desbloqueo de señales:**
 - Después de 10 segundos, se restaura la máscara original con **sigprocmask**, permitiendo que la señal **SIGTERM** pendiente (si existe) sea procesada.
- 5. **Verificación de la señal:**
 - Se verifica si **SIGTERM** fue recibida utilizando la bandera **signal_recibida**. Si fue recibida, se imprime un mensaje.

Ejecución:

gcc -o tarea12.c -o t12

```
tomy@Lenovo-Tomas:~/Documentos/2ºCARRERA/S0/ModuloII/Sesión $ ./t12 &
[1] 14065
tomy@Lenovo-Tomas:~/Documentos/2ºCARRERA/S0/ModuloII/Sesión $ kill -SIGTERM 14065
tomy@Lenovo-Tomas:~/Documentos/2ºCARRERA/S0/ModuloII/Sesión $ kill -SIGTERM 14065
tomy@Lenovo-Tomas:~/Documentos/2ºCARRERA/S0/ModuloII/Sesión $ kill -SIGTERM 14065
tomy@Lenovo-Tomas:~/Documentos/2ºCARRERA/S0/ModuloII/Sesión $
Señal recibida
kill -SIGTERM 14065
bash: kill: (14065) - No existe el proceso
[1]+  Hecho                ./t12
tomy@Lenovo-Tomas:~/Documentos/2ºCARRERA/S0/ModuloII/Sesión $
```

Si en 10 segundos se ha recibido la señal, finaliza.