UNIVERSIDAD AUTÓNOMA DE BAJA CALIFORNIA
FACULTAD DE INGENIERÍA (UNIDAD MEXICALI)

Formato para prácticas de laboratorio

PROGRAMA EDUCATIVO	PLAN DE ESTUDIO	CLAVE DE UNIDAD DE APRENDIZAJE	NOMBRE DE LA UNIDAD DE APRENDIZAJE
Ingeniero en Computación	2009-2	12124	Taller de Sistema Operativo Unix

PRÁCTICA No.	LABORATORIO DE	Ingeniero en Computación	DURACIÓN (HORAS)
6	**NOMBRE DE LA PRÁCTICA**	Mandos para el manejo de procesos	2

1. INTRODUCCIÓN

Un proceso es una instancia de un programa. En UNIX, un proceso se crea cuando se invoca o ejecuta un programa, ya sea mediante un mando del shell o por medio de otro programa. En la presente práctica se conocerán y utilizarán los principales mandos con los que cuenta UNIX para el manejo de los procesos.

2. OBJETIVO (COMPETENCIA)

El alumno utilizará los mandos de UNIX para supervisar y manipular la ejecución de los procesos en el sistema.

3. FUNDAMENTO

Se le llama proceso en Unix a un programa en ejecución y al objeto abstracto que crea el sistema operativo para manejar el acceso de ese programa a los recursos del sistema (memoria, CPU,

Formuló	Revisó	Aprobó	Autorizó
Linda Eugenia Arredondo Acosta José Martín Olguín Espinoza Eva Herrera Ramírez	Aglay González Pacheco		David I. Rosas Almeida
Nombre y Firma del Maestro	Nombre y Firma del Responsable de Programa Educativo	Nombre y Firma del Responsable de Gestión de Calidad	Nombre y Firma del Director / Representante de la Dirección

dispositivos de E/S). Pueden coexistir varias instancias de un mismo programa ejecutando en forma simultánea. Cada una de ellas es un proceso diferente.

En Unix, a los ojos de un usuario en un momento dado hay múltiples programas en ejecución, cada uno de ellos avanzando en su tarea. Sin embargo en una máquina con un solo procesador hay en cada instante solamente un proceso ejecutando. Es el sistema operativo el que va rotando el uso del procesador a intervalos breves (alguna decena de milisegundos) entre los procesos definidos en el sistema de forma que se crea la ilusión que todos avanzan simultáneamente.

El administrador del sistema dispone de herramientas para supervisar el estado de los procesos y eventualmente tomar acciones para suspender o detener la ejecución de un proceso o simplemente modificar su comportamiento.

Para cada proceso definido en el sistema, el *kernel* (núcleo) del sistema operativo almacena y mantiene al día varios tipos de información sobre el proceso. Esta información podemos ordenarla de la siguiente forma:

- Información general. Identificadores de proceso, usuario y grupo
- Ambiente (variables, directorio actual, etc.)
- Información de E/S
- Información de estado
- Espacio de direcciones del proceso. Áreas de trabajo (código, datos, stack)

Identificadores

Process ID (PID)

Al crearse un nuevo proceso se le asigna un identificador de proceso único. Este número debe utilizarse por el administrador para referirse a un proceso dado al ejecutar un mando. Los PID son asignados por el sistema a cada nuevo proceso en orden creciente comenzando desde cero.

Parent Process ID (PPID)

Los procesos a su vez pueden crear subprocesos, llamados procesos hijo (child process), un proceso puede tener varios procesos hijo, pero un proceso hijo sólo tendrá un padre. El PPID de un proceso es el PID de su proceso padre.

UID y EUID

El *User ID* (UID) del proceso identifica al creador del proceso, esto es a la persona que inició la ejecución. Este usuario y root son los únicos que pueden modificar al proceso.

El *Effective User ID* (EUID) en cambio se utiliza para determinar si el proceso tiene permiso para acceder a archivos y otros recursos del sistema.

GID y EGID

El GID es totalmente análogo a los identificadores de usuario pero para grupos de usuarios. El GID se hereda del proceso padre. El EGID puede utilizarse igual que el EUID para controlar el acceso del proceso a archivos.

Supervisión de procesos

Mando ps (process status)

La herramienta básica para diagnosticar problemas relacionados con procesos es el mando **ps**. Este mando muestra información acerca de los procesos activos.

Si se ejecuta **ps** sin parámetros solamente se listará información básica (muestra el número del proceso, terminal, tiempo acumulado de ejecución y el nombre del mando) sobre los procesos que pertenecen a mi usuario.

```
tallerunix@linda-VirtualBox:/home/linda$ ps
  PID TTY          TIME CMD
 3467 pts/0    00:00:00 su
 3475 pts/0    00:00:00 bash
 3745 pts/0    00:00:00 ps
```

Figura 1: Salida del mando **ps** sin opciones

A través del uso de parámetros adecuados se puede mostrar más información sobre cada proceso. Un conjunto de parámetros que permite ver todos los procesos con más detalle son:

ps uax

```
larredondo@computacion:~$ ps aux
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.0  24452  2404 ?        Ss   Aug31   0:01 /sbin/init
root         2  0.0  0.0      0     0 ?        S    Aug31   0:00 [kthreadd]
root         3  0.0  0.0      0     0 ?        S    Aug31   0:14 [ksoftirqd/0]
root         6  0.0  0.0      0     0 ?        S    Aug31   0:00 [migration/0]
root         7  0.0  0.0      0     0 ?        S    Aug31   0:01 [watchdog/0]
root         8  0.0  0.0      0     0 ?        S    Aug31   0:00 [migration/1]
root        10  0.0  0.0      0     0 ?        S    Aug31   0:00 [ksoftirqd/1]
root        12  0.0  0.0      0     0 ?        S    Aug31   0:01 [watchdog/1]
root        13  0.0  0.0      0     0 ?        S    Aug31   0:00 [migration/2]
root        15  0.0  0.0      0     0 ?        S    Aug31   0:00 [ksoftirqd/2]
root        16  0.0  0.0      0     0 ?        S    Aug31   0:01 [watchdog/2]
root        17  0.0  0.0      0     0 ?        S    Aug31   0:00 [migration/3]
```

Figura 2: Salida, recortada, del mando **ps aux**

Los campos de información más importantes desplegados por **ps** para cada proceso son:

PID	Process ID, número único o de identificación del proceso.
PPID	Parent Process ID, padre del proceso
USER	User ID, usuario propietario del proceso
TTY	Terminal asociada al proceso, si no hay terminal aparece entonces un '?'
TIME	Tiempo de uso de cpu acumulado por el proceso
COMMAND	El nombre del programa o mando que inició el proceso
RSS	Resident Size, tamaño de la parte residente en memoria en kilobytes
SZ o SIZE	Tamaño virtual de la imagen del proceso
%CPU	Porcentaje de cpu utilizado por el proceso
TIME	Starting Time, hora de inicio del proceso
STAT	Status del proceso, estos pueden ser los siguientes

- **R** *runnable*, en ejecución, corriendo o ejecutándose

- **S** *sleeping*, proceso en ejecución pero sin actividad por el momento, o esperando por algún evento para continuar
- **T** *sTopped*, proceso detenido totalmente, pero puede ser reiniciado
- **Z** *zombie*, difunto, proceso que por alguna razón no terminó de manera correcta, no debe haber procesos zombies
- **D** *uninterruptible* sleep, son procesos generalmente asociados a acciones de IO del sistema
- **X** *dead*, muerto, proceso terminado pero que sigue apareciendo, igual que los **Z** no deberían verse nunca

La opción **-G [grupo]** muestra los procesos por grupo.

```
larredondo@computacion:~$ ps -G maestros
  PID TTY          TIME CMD
 1083 ?        00:00:00 sshd
 1084 pts/1    00:00:00 bash
 1593 pts/1    00:00:00 ps
```

Figura 3: Muestra los procesos del grupo maestros

Con la opción **-U [usuario]** se pueden visualizar los procesos del usuario, por ejemplo en la siguiente imagen podemos ver los procesos del usuario **`tallerunix`**:

```
tallerunix@linda-VirtualBox:/home/linda$ ps -U tallerunix
  PID TTY          TIME CMD
 2504 pts/0    00:00:00 su
 2512 pts/0    00:00:00 bash
 2567 pts/0    00:00:00 ps
```

Figura 4: Salida del mando **ps** con opción **U**

-f Muestra un listado de formato completo, como el que se muestra en la figura 5:

```
tallerunix@linda-VirtualBox:/home/linda$ ps -f
UID        PID  PPID  C STIME TTY          TIME CMD
1002      2504  1954  0 19:54 pts/0    00:00:00 su tallerunix
1002      2512  2504  0 19:54 pts/0    00:00:00 bash
1002      2576  2512  0 20:01 pts/0    00:00:00 ps -f
```

Figura 5: Salida del mando **ps** con opción **f**

Recuerde que también podemos utilizar una combinación de opciones, como se aprecia en la figura 6, en donde vemos los procesos activos del administrador del sistema (usuario root) en formato completo:

```
larredondo@computacion:~$ ps -F -U root
UID        PID  PPID  C    SZ   RSS PSR STIME TTY          TIME CMD
root         1     0  0  6113  2404   0 Aug31 ?        00:00:01 /sbin/init
root         2     0  0     0     0   3 Aug31 ?        00:00:00 [kthreadd]
root         3     2  0     0     0   0 Aug31 ?        00:00:14 [ksoftirqd/0]
root         6     2  0     0     0   0 Aug31 ?        00:00:00 [migration/0]
root         7     2  0     0     0   0 Aug31 ?        00:00:01 [watchdog/0]
root         8     2  0     0     0   1 Aug31 ?        00:00:00 [migration/1]
root        10     2  0     0     0   1 Aug31 ?        00:00:00 [ksoftirqd/1]
root        12     2  0     0     0   1 Aug31 ?        00:00:01 [watchdog/1]
root        13     2  0     0     0   2 Aug31 ?        00:00:00 [migration/2]
root        15     2  0     0     0   2 Aug31 ?        00:00:00 [ksoftirqd/2]
root        16     2  0     0     0   2 Aug31 ?        00:00:01 [watchdog/2]
root        17     2  0     0     0   3 Aug31 ?        00:00:00 [migration/3]
root        19     2  0     0     0   3 Aug31 ?        00:00:00 [ksoftirqd/3]
```

Figura 6: Salida, recortada, del mando **ps -F -U root**

Si desea visualizar las opciones completas del mando **ps** puede utilizar el manual de ayuda (**man ps**) y para ver un resumen de sus opciones más comunes use **ps --help**:

```
tallerunix@linda-VirtualBox:/home/linda$ ps --help
********* simple selection *********  ********* selection by list *********
-A all processes                      -C by command name
-N negate selection                   -G by real group ID (supports names)
-a all w/ tty except session leaders  -U by real user ID (supports names)
-d all except session leaders         -g by session OR by effective group name
-e all processes                      -p by process ID
T  all processes on this terminal     -s processes in the sessions given
a  all w/ tty, including other users  -t by tty
g  OBSOLETE -- DO NOT USE             -u by effective user ID (supports names)
r  only running processes             U  processes for specified users
x  processes w/o controlling ttys     t  by tty
*********** output format **********  *********** long options ***********
-o,o user-defined  -f full            --Group --User --pid --cols --ppid
-j,j job control   s  signal          --group --user --sid --rows --info
-O,O preloaded -o  v  virtual memory  --cumulative --format --deselect
-l,l long          u  user-oriented   --sort --tty --forest --version
-F   extra full    X  registers       --heading --no-heading --context
                     ********* misc options *********
-V,V  show version      L  list format codes  f  ASCII art forest
-m,m,-L,-T,H  threads   S  children in sum    -y change -l format
-M,Z  security data     c  true command name  -c scheduling class
-w,w  wide output       n  numeric WCHAN,UID  -H process hierarchy
```

Figura 7: Salida del mando **ps** con la opción help

Mando top

El mando **ps** muestra todos los procesos en un momento determinado, pero si quieres verlos en tiempo real de ejecución puedes utilizar el mando **top** y te saldrá una imagen similar a la figura 8:

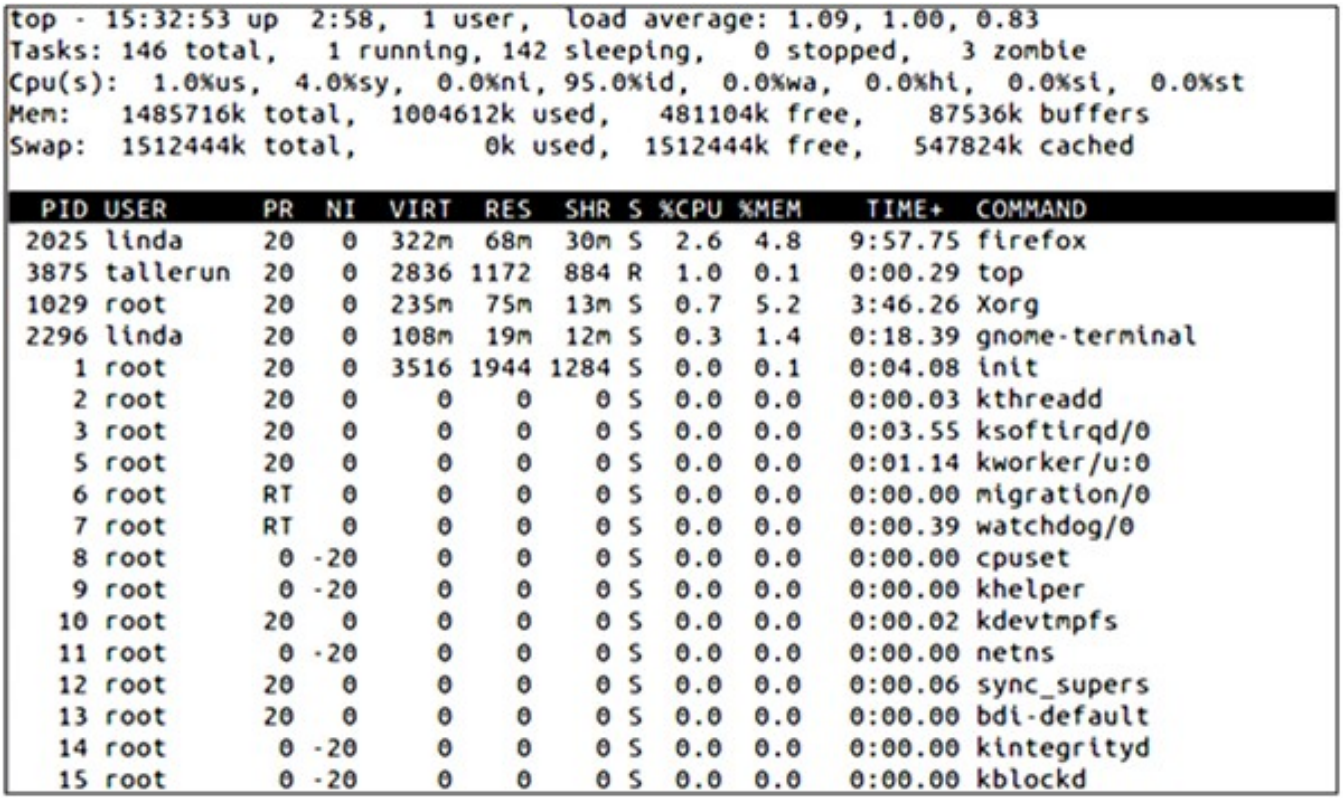

Figura 8: Salida en pantalla del mando **top**

En la figura 8 podemos apreciar la siguiente información:

- En el primer renglón de la cabecera, antes de la línea negra, tenemos la hora, la cantidad de usuarios (en este caso 1) y el promedio de carga del procesador
- En el segundo renglón nos indica la cantidad de tareas o procesos que hay en ejecución y cuantos separados de acuerdo al estado de los mismos. En el ejemplo tenemos 146 en total de los cuales 142 están en espera, ninguno está detenido y hay 3 procesos zombie
- El tercer renglón muestra el estado del cpu
- Después tenemos las información de la memoria del sistema, en este caso hay un total de 1485716k de los cuales 1004612 están en uso y 481104k estan libres
- El siguiente renglón muestra la información de la memoria swap
- En el cuerpo del reporte se muestra la información de los procesos activos, en donde vemos el número del proceso (PID), el usuario que lanzó el proceso (User), prioridad de ejecucion del proceso (PR), memoria virtual (VIRT), memoria compartida (SHR), el estado del proceso (S), el porcentaje de uso de CPU, el porcentaje de uso de memoria, el nombre del proceso (COMMAND),

Cuando quiera salir de la aplicación pulse **q**.

Manipulación de procesos

Procesos en primer plano (Foreground)

Cuando se ingresa un mando, el shell crea un proceso hijo en el cual se ejecuta el mando. El proceso padre (el shell en este caso) se queda esperando a que se termine de ejecutar este proceso hijo. Mientras se esté ejecutando, el usuario no puede teclear otro mando, se dice entonces que el proceso hijo está ejecutándose en el primer plano.

Una vez que el mando termina de ejecutarse, el proceso hijo muere, el proceso padre se reactiva, la entrada de la línea de mandos (shell prompt) aparece y entonces se puede teclear otro mando. En resumen, un proceso que está ejecutándose en el primer plano es aquél que tiene la atención total de la terminal.

Tareas en segundo plano (Background Jobs)

Un proceso que se inicie desde la línea de mandos se puede ejecutar en el segundo plano como una tarea (*job*) y así evitar que se bloquee la terminal. Para hacer esto sólo se requiere agregar un símbolo **&** (*ampersand*) al final del mando. Por ejemplo:

```
tallerunix@linda-VirtualBox:/home/linda$ ./practica7 &
[1] 2986
tallerunix@linda-VirtualBox:/home/linda$ bash: ./practica7:
```

Figura 9: Ejecución de un proceso en segundo plano

En el ejemplo anterior, el programa ***practica7*** fue invocado desde la línea de mandos con la instrucción de que se ejecute en segundo plano. El sistema operativo UNIX inicia el proceso y retorna el PID que le asignó así como el número de tarea que le asignó entre corchetes [ ].

Si desplegamos la información de los procesos, nos mostrará algo parecido a lo siguiente:

```
linda@linda-VirtualBox:~$ ps
  PID TTY          TIME CMD
 3097 pts/0    00:00:00 bash
 3158 pts/0    00:00:00 sleep
 3166 pts/0    00:00:00 practica7
 3167 pts/0    00:00:00 sleep
 3168 pts/0    00:00:00 ps
linda@linda-VirtualBox:~$ jobs
[1]+  Running                 ./practica7 &
linda@linda-VirtualBox:~$
```

Figura 10: Revisión de procesos o tareas en segundo plano

Y para saber cuáles procesos están ejecutándose en segundo plano utilizamos el mando **jobs,** como en el ejemplo anterior en donde se encuentra en ejecución el programa denominado ***practica7***. En la figura 10 se aprecia que podemos seguir ejecutando mandos en la misma terminal en donde lanzamos la ejecucion en segundo plano del programa ***practica7***.

Si deseamos cambiar la ejecución del programa ***practica7*** del segundo al primer plano, utilizamos el mando **fg**, como se muestra a continuación:

```
linda@linda-VirtualBox:~$ jobs
[1]+  Running                 ./practica7 &
linda@linda-VirtualBox:~$ fg 1
./practica7
```

Figura 11: Cambio de proceso de segundo a primer plano

Para poner en segundo plano un proceso que se está ejecutando en primer plano es necesario realizar los siguientes pasos:

1. Suspender el proceso presionando la combinación de la tecla Ctrl y la tecla z (Ctrl-z)
2. Teclear el mando **bg** para enviar el proceso al background.

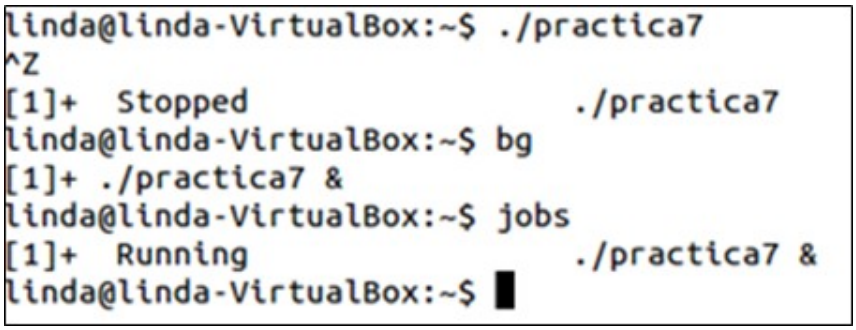

Figura 12: Ejemplo para enviar un proceso a ejecución en segundo plano

Terminando procesos

Algunas veces, por algún error de operación, los procesos no terminan correctamente y es necesario terminarlos de alguna manera. El procedimiento usual en estos casos es obtener el PID del proceso con la ayuda del mando **ps** y luego terminarlo con el comando **kill**, que se utiliza para enviar una señal al proceso. Linux soporta varias señales estándar, en la tabla 1 vemos algunas del estándar POSIX. 1:

SEÑAL	VALOR	ACCIÓN	DESCRIPCIÓN
SIGHUP	1	Term	Error en la terminal de control o muerte del proceso de control
SIGINT	2	Term	Interrupción procedente del teclado
SIGQUIT	3	Core	Terminación procedente del teclado
SIGKILL	9	Term	Señal de matar
SIGSEGV	11	Core	Referencia inválida a memoria
SIGALRM	14	Term	Señal de alarma
SIGTERM	15	Term	Señal de terminación
SIGUSR1	30,10,16	Term	Señal definida por usuario 1
SIGUSR2	31,12,17	Term	Señal definida por usuario 2
SIGCHLD	20,17,18	Ign	Proceso hijo terminado o detenido
SIGCONT	19,18,25		Continuar si estaba detenido
SIGSTOP	17,19,23	Stop	Detener el proceso

Tabla 1: Algunas de las señales de POSIX. 1

Los términos en la columna ACCIÓN de la tabla 1 especifican la acción por defecto para la señal de la siguiente manera:

Term	Terminar el proceso.
Ign	Ignorar la señal.
Core	Terminar el proceso y realizar un desbordamiento de memoria.
Stop	Detener el proceso.

Las señales **SIGKILL** y **SIGSTOP** no pueden ser capturadas, bloqueadas o ignoradas.

También puede suceder que algún proceso quede fuera de control y comience a acaparar algún recurso del sistema (memoria, disco o CPU). Esto puede suceder por un error de programación, por un error de configuración, por intentar ejecutar algún proceso que necesita más recursos que los disponibles.

Sea cual sea el caso se hace necesario que el dueño del proceso o el administrador del sistema tomen medidas para frenar o terminar a ese proceso. El primer paso será identificar el o los procesos problemáticos utilizando las herramientas ya vistas.

Las siguientes opciones son solicitar al proceso que termine enviando la señal **TERM,** a la cual le corresponde el número 15 de acuerdo con la tabla 1, (**kill -15 PID**):

```
linda@linda-VirtualBox:~$ ps
  PID TTY          TIME CMD
 1954 pts/0    00:00:00 bash
 2625 pts/0    00:00:00 su
 2633 pts/0    00:00:00 bash
 2689 pts/0    00:00:00 ps
linda@linda-VirtualBox:~$ kill -15 2625

Session terminated, terminating shell...linda@linda-VirtualBox:~$  ...killed.
tallerunix@linda-VirtualBox:/home/linda$
```

Figura 13: Ejemplo de solicitud de terminación de un proceso

En la figura 13 se aprecia que el mando **ps** se ejecuta con el usuario **`linda`** y al enviar la señal **TERM** al proceso **2625**, se termina la sesión del usuario **`linda`** y regresa el shell al usuario **`tallerunix`**.

También se tiene la opción de terminar el proceso por la fuerza con la señal **KILL,** a la cual le corresponde el número 9 de acuerdo con la tabla 1, (**kill -9 PID**)

```
tallerunix@linda-VirtualBox:/home/linda$ ps
  PID TTY          TIME CMD
 2779 pts/0    00:00:00 su
 2787 pts/0    00:00:00 bash
 2842 pts/0    00:00:00 ps
tallerunix@linda-VirtualBox:/home/linda$ kill -9 2779
Killed
tallerunix@linda-VirtualBox:/home/linda$ linda@linda-VirtualBox:~$ exit

linda@linda-VirtualBox:~$
```

Figura 14: Terminando un proceso

Otra situación que puede ocurrir es que al listar los procesos activos el que se desea terminar tiene distintas instancias abiertas, es decir, dispone de varias **PID** y para terminarlo sería necesario utilizar varias veces el mando **kill.** El mando **killall** se utiliza para finalizar todos los procesos que abre un comando. Como se puede suponer, al disponer de distintas **PID** no es ésta la que se le debe indicar al

comando **killall** sino el nombre del proceso. Este nombre viene dado en el listado de procesos como **CMD** o **COMMAND**; entonces:

```
larredondo@computacion:~$ ./practica7 &
[1] 2012
larredondo@computacion:~$ ./practica7 &
[2] 2014
larredondo@computacion:~$ ./practica7 &
[3] 2016
larredondo@computacion:~$ ps
  PID TTY          TIME CMD
 1084 pts/1    00:00:00 bash
 2012 pts/1    00:00:00 practica7
 2013 pts/1    00:00:00 sleep
 2014 pts/1    00:00:00 practica7
 2015 pts/1    00:00:00 sleep
 2016 pts/1    00:00:00 practica7
 2017 pts/1    00:00:00 sleep
 2018 pts/1    00:00:00 ps
larredondo@computacion:~$ killall practica7
[1]   Terminated              ./practica7
[2]-  Terminated              ./practica7
[3]+  Terminated              ./practica7
larredondo@computacion:~$ ps
  PID TTY          TIME CMD
 1084 pts/1    00:00:00 bash
 2013 pts/1    00:00:00 sleep
 2015 pts/1    00:00:00 sleep
 2017 pts/1    00:00:00 sleep
 2020 pts/1    00:00:00 ps
```

Figura 15: Ejemplo del uso del mando **killall**

El ejemplo de la figura 15 se realizó de la siguiente manera: primero se ejecutó, en segundo plano, tres veces el programa *practica7*, al ver el listado de procesos se muestra 3 veces el proceso *practica7* pero con diferente número de identificador. Entonces, se utiliza el mando **killall practica7** para eliminar, al mismo tiempo, todas las instancias de dicho proceso. Al mostrar de nuevo el listado de procesos ya no encontramos a *practica7*

4. PROCEDIMIENTO (DESCRIPCIÓN)

A) EQUIPO NECESARIO	**MATERIAL DE APOYO**

Computadora con Linux Ubuntu instalado. Páginas de ayuda en línea (man) para los mandos **ps**, **top**,

bg, killall y **kill.**

B) DESARROLLO DE LA PRÁCTICA

Para cada paso de la práctica anota los mandos que requieres:

1. Inicia una sesión con tu cuenta en el servidor computación.
2. Lista todos los procesos activos en el sistema.
3. Lista los procesos que está ejecutando el grupo de trabajo al que perteneces.
4. Lista sólo los procesos que te pertenecen.
5. Pon en funcionamiento en segundo el programa *practica7* ubicado en el directorio */externos/home/clases/compartido*
6. Anota el número de trabajo.
7. Lista los procesos que está ejecutando el grupo de trabajo al que perteneces utilizando la opción de formato orientado al usuario.
8. Cambia el proceso *practica7* del segundo al primer plano.
9. Envía de nuevo el proceso al segundo plano.
10. Lista los procesos de tres de tus compañeros.
11. Trata de eliminar los procesos de tus compañeros. ¿Qué pasa? ¿Por qué?
12. Lista los jobs.
13. Solicita que termine la ejecución del proceso *practica7*.
14. Verifica que el proceso *practica7* ya no existe.
15. Pon nuevamente en ejecución el programa *practica7* en segundo plano.
16. Ahora ejecuta el programa *procesos* en el primer plano.
17. Mandalo a ejecución en el segundo plano.
18. Ejecuta de nuevo el programa *practica7* en el segundo plano.
19. Visualiza las tareas (*jobs)* que se están ejecutando en segundo plano.
20. Elimina, en un solo paso, los procesos denominados *practica7* que se están ejecutando en segundo plano.
21. Termina todos los procesos que se estén ejecutando en segundo plano

C) CÁLCULOS (SI APLICA) Y REPORTE

5. RESULTADOS Y CONCLUSIONES

6. ANEXOS

7. REFERENCIAS

http://www.ubuntu.com
http://www.linuxtotal.com.mx/
http://viviendoentrepinguinos.wordpress.com/2010/04/16/los-procesos-en-linux/
http://www.rinconinformatico.net/monitorear-procesos-en-gnulinux/