

UNIVERSIDAD RAFAEL LANDÍVAR

FACULTAD DE INGENIERÍA

SISTEMAS OPERATIVOS

SECCIÓN 1 VESPERTINA

ING. JULIO REQUENA

PRACTICA 2

Julio Anthony Engels Ruiz Coto 1284719

GUATEMALA DE LA ASUNCIÓN, FEBRERO 20 DE 2023

CAMPUS CENTRAL

1) Ejecute el comando top que permita monitorizar únicamente los procesos por un determinado ID, que permite visualizar la línea de comandos completa del proceso y que permita refrescar la información hasta 10 veces.

top -p 2166 -c -d 1 -n 10

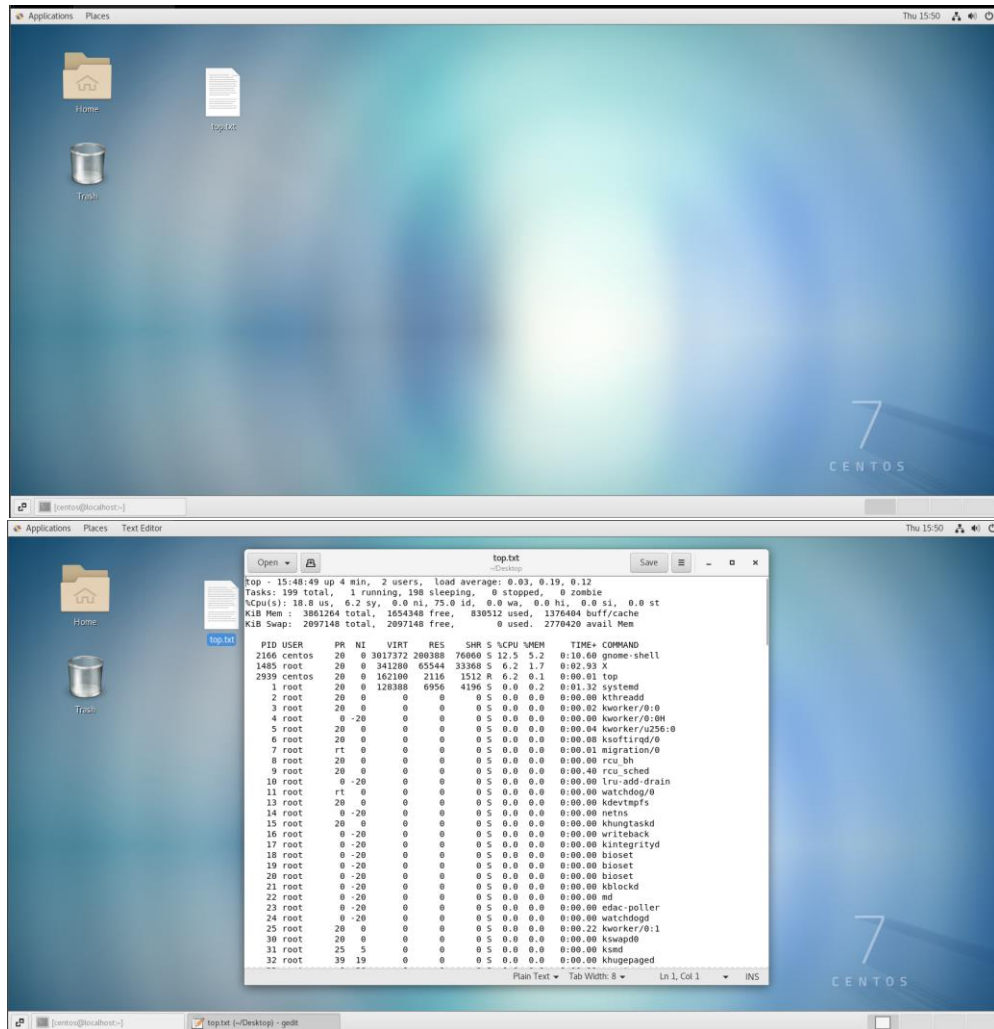
```
top - 15:55:14 up 11 min, 2 users, load average: 0.00, 0.05, 0.08
Tasks: 1 total, 0 running, 1 sleeping, 0 stopped, 0 zombie
%Cpu(s): 3.0 us, 0.0 sy, 0.0 ni, 97.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
KiB Mem : 3861264 total, 1636096 free, 842312 used, 1382856 buff/cache
KiB Swap: 2097148 total, 2097148 free, 0 used, 2756284 avail Mem

  PID USER      PR  NI  VIRT  RES  SHR  S  %CPU  %MEM    TIME+  COMMAND
 2166 centos    20   0 3817620 200836 76136 S   2.0   5.2   0:18.74 /usr/bin/gnome-shell
```

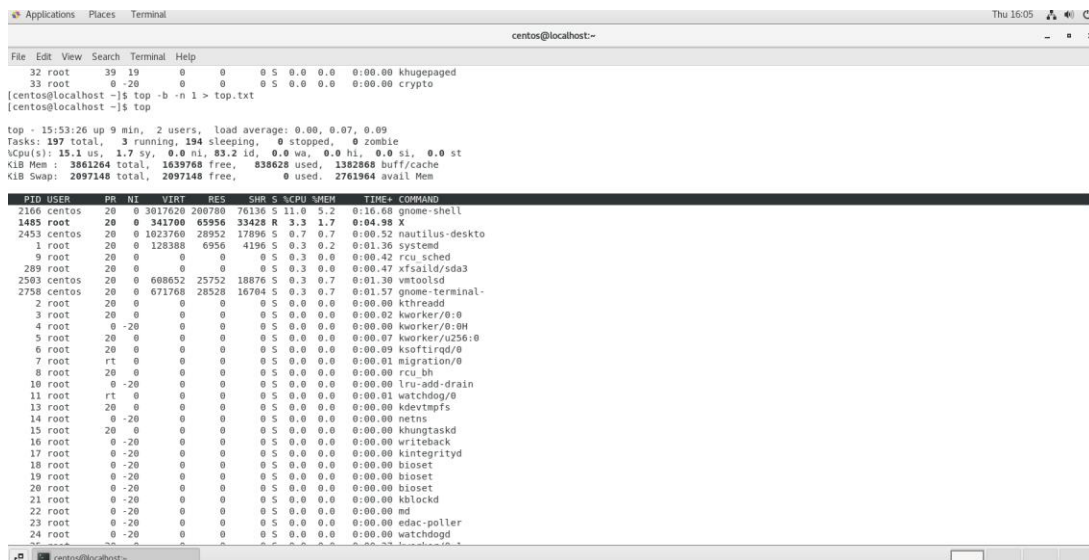
2) Investigue cómo puede ordenar campos utilizando el comando top y almacene el resultado en el archivo denominado top.txt.

```
top - 15:58:11 up 6 min, 2 users, load average: 0.01, 0.14, 0.11
Tasks: 196 total, 2 running, 194 sleeping, 0 stopped, 0 zombie
%Cpu(s): 10.0 us, 1.0 sy, 0.0 ni, 88.7 id, 0.0 wa, 0.0 hi, 0.3 si, 0.0 st
KiB Mem : 3861264 total, 1641688 free, 836864 used, 1382792 buff/cache
KiB Swap: 2097148 total, 2097148 free, 0 used, 2756324 avail Mem

  PID USER      PR  NI  VIRT  RES  SHR  S  %CPU  %MEM    TIME+  COMMAND
 2166 centos    20   0 3817620 201060 76132 S   7.3   5.2   0:14.74 gnome-shell
1485 root        20   0 341780 66832 33428 S   3.3   1.7   0:04.39 X
2758 centos    20   0 871768 28528 16784 S   1.0   0.7   0:01.40 gnome-terminal-
1061 root        20   0 574292 19536 6152 S   0.3   0.5   0:00.22 tuned
2453 centos    20   0 1831956 28940 17892 S   0.3   0.7   0:00.37 nautilus-desktop
2563 centos    20   0 608524 25532 18656 S   0.3   0.7   0:00.04 vtgostd
2957 centos    20   0 162108 2340 1572 R   0.3   0.1   0:00.12 top
  1 root        20   0 128388 6956 4196 S   0.0   0.2   0:01.34 systemd
  2 root        20   0 0 0 0 S   0.0   0.0   0:00.00 kthreadd
  3 root        20   0 0 0 0 S   0.0   0.0   0:00.02 kworker/0:0
  4 root        20   0 0 0 0 S   0.0   0.0   0:00.00 kworker/0:0H
  5 root        20   0 0 0 0 S   0.0   0.0   0:00.04 kworker/u256:0
  6 root        20   0 0 0 0 S   0.0   0.0   0:00.00 ksoftirqd/0
  7 root        rt    0 0 0 0 S   0.0   0.0   0:00.01 migration/0
  8 root        20   0 0 0 0 S   0.0   0.0   0:00.00 rcu_bh
  9 root        20   0 0 0 0 S   0.0   0.0   0:00.41 rcu_sched
10 root        20   0 0 0 0 S   0.0   0.0   0:00.00 lru-add-drain
11 root        rt    0 0 0 0 S   0.0   0.0   0:00.00 watchdog/0
13 root        20   0 0 0 0 S   0.0   0.0   0:00.00 kdevtmpfs
14 root        20   0 0 0 0 S   0.0   0.0   0:00.00 netns
15 root        20   0 0 0 0 S   0.0   0.0   0:00.00 khungtaskd
16 root        20   0 0 0 0 S   0.0   0.0   0:00.00 writeback
17 root        20   0 0 0 0 S   0.0   0.0   0:00.00 kintegrityd
18 root        20   0 0 0 0 S   0.0   0.0   0:00.00 bioset
19 root        20   0 0 0 0 S   0.0   0.0   0:00.00 bioset
20 root        20   0 0 0 0 S   0.0   0.0   0:00.00 bioset
21 root        20   0 0 0 0 S   0.0   0.0   0:00.00 kblockd
22 root        20   0 0 0 0 S   0.0   0.0   0:00.00 md
23 root        20   0 0 0 0 S   0.0   0.0   0:00.00 edac-poller
24 root        20   0 0 0 0 S   0.0   0.0   0:00.00 watchdogd
25 root        20   0 0 0 0 S   0.0   0.0   0:00.25 kworker/0:1
```

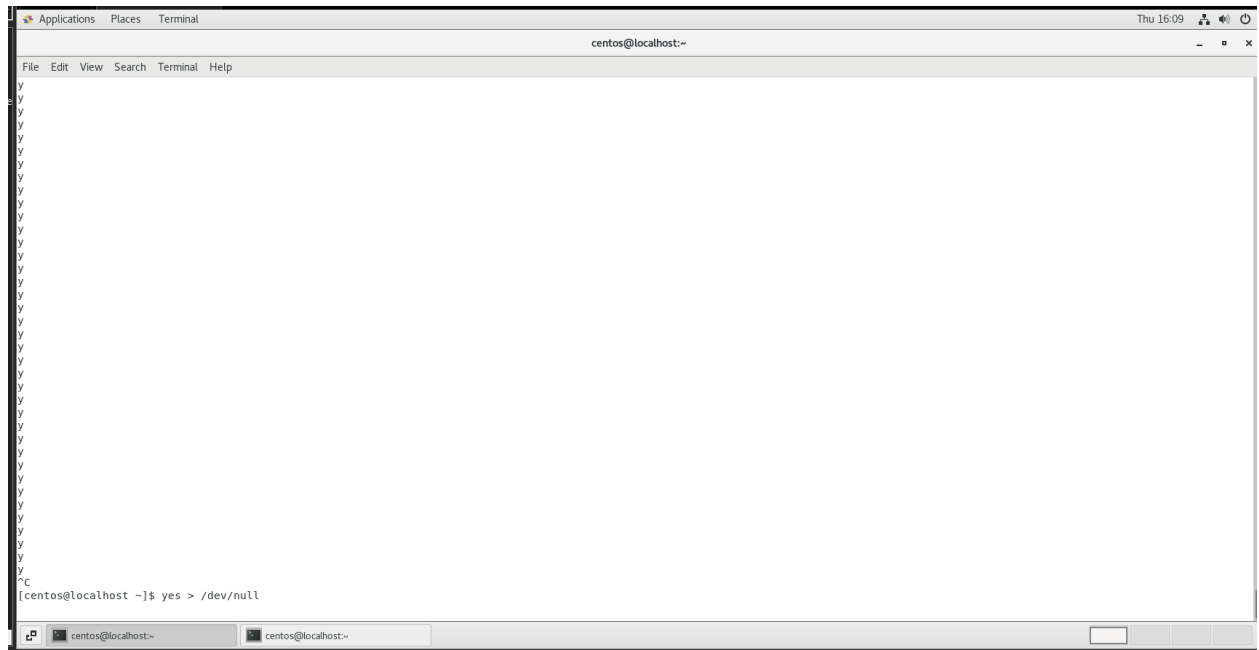


3) Cambie a la prioridad máxima todos los procesos de un determinado usuario.



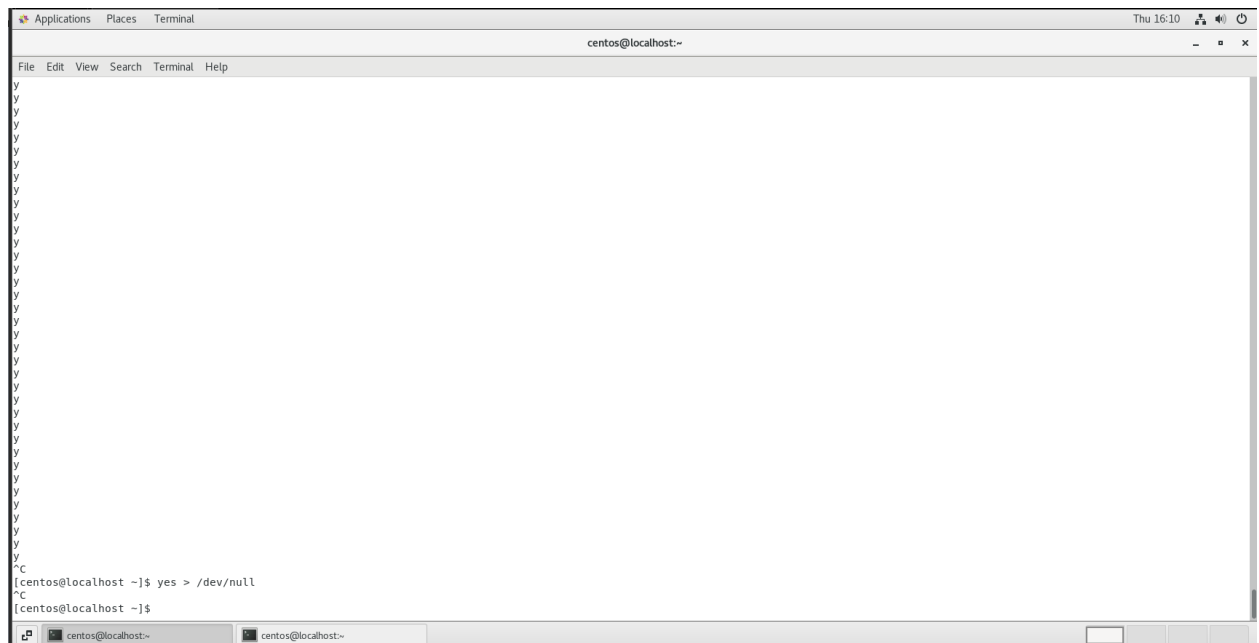
3) Vuelve a ejecutarlo redirigiendo la salida a /dev/null. Comprueba que ahora la pantalla no se ensucia, pero el prompt sigue sin aparecer.

yes > /dev/null



4) Elimínalo otra vez con CTRL+C.

Ctrl+c

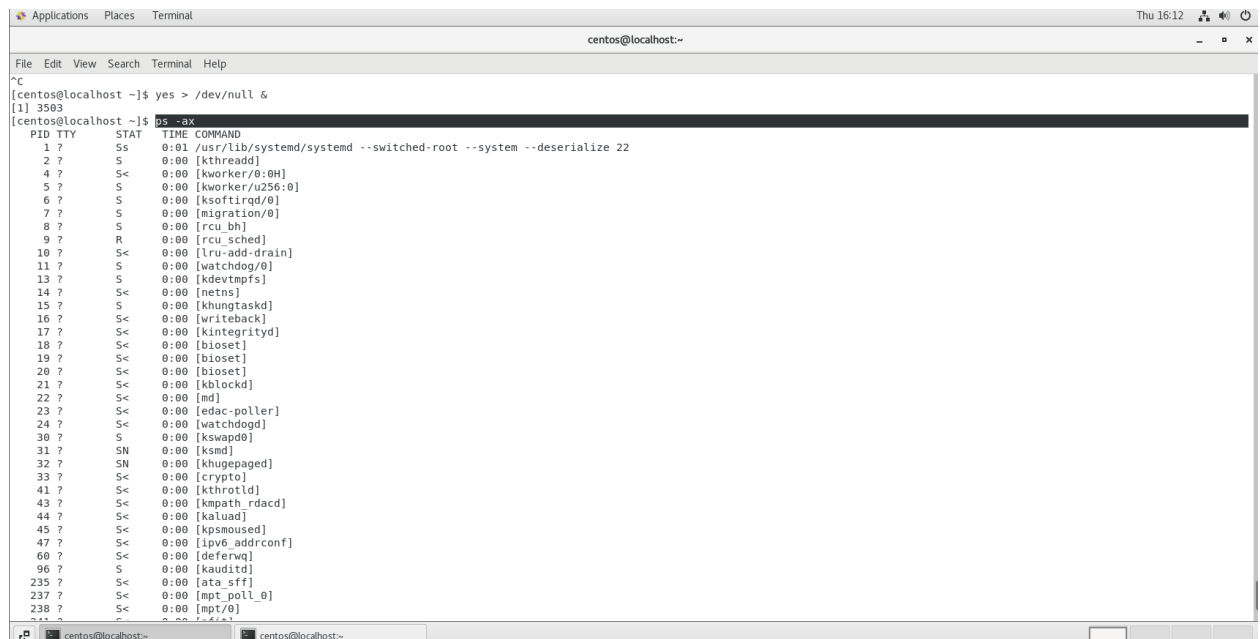


5) Vuelve a ejecutarlo redirigiendo la salida y añadiendo al final el carácter &. Comprueba que ahora la shell te permite seguir trabajando. Anota el número de tarea y el PID del proceso.

Yes > /dev/null &

Ps -ax

PID 1156



```
~C
[centos@localhost ~]$ yes > /dev/null &
[1] 3503
[centos@localhost ~]$ ps -ax
PID TTY STAT TIME COMMAND
1 ? Ss 0:01 /usr/lib/systemd/systemd --switched-root --system --deserialize 22
2 ? S 0:00 [kthreadd]
4 ? S< 0:00 [kworker/0:0H]
5 ? S 0:00 [kworker/u256:0]
6 ? S 0:00 [ksoftirqd/0]
7 ? S 0:00 [migration/0]
8 ? S 0:00 [rcu_bh]
9 ? R 0:00 [rcu_sched]
10 ? S< 0:00 [lru-add-drain]
11 ? S 0:00 [watchdog/0]
13 ? S 0:00 [kdevtmpfs]
14 ? S< 0:00 [netns]
15 ? S 0:00 [khungtaskd]
16 ? S< 0:00 [writeback]
17 ? S< 0:00 [kintegrityd]
18 ? S< 0:00 [bioset]
19 ? S< 0:00 [bioset]
20 ? S< 0:00 [bioset]
21 ? S< 0:00 [kblockd]
22 ? S< 0:00 [md]
23 ? S< 0:00 [edac-poller]
24 ? S< 0:00 [watchdogd]
30 ? S 0:00 [kswapd0]
31 ? SN 0:00 [ksmd]
32 ? SN 0:00 [khugepaged]
33 ? S< 0:00 [crypto]
41 ? S< 0:00 [kthrotld]
43 ? S< 0:00 [kmpath_rdad]
44 ? S< 0:00 [kaluad]
45 ? S< 0:00 [kpsmouse]
47 ? S< 0:00 [ipv6_addrconf]
60 ? S< 0:00 [deferwq]
96 ? S 0:00 [kauditd]
235 ? S< 0:00 [ata_sff]
237 ? S< 0:00 [mpt_poll_0]
238 ? S< 0:00 [mpt/0]
```

6) Ejecuta el comando jobs para ver el estado del proceso que se está ejecutando.

Jobs

[1] Ejecutando yes > /dev/null &



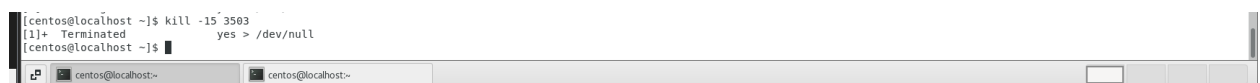
```
[centos@localhost ~]$ jobs
[1]+  Running                  yes > /dev/null &
[centos@localhost ~]$
```

7) Elimina el proceso asociado a la tarea con el comando kill, y comprueba con el comando jobs que realmente ha finalizado.

Kill -15 1156

Jobs

Terminado



```
[centos@localhost ~]$ kill -15 3503
[1]+  Terminated              yes > /dev/null
[centos@localhost ~]$
```

8) Vuelve a lanzar el proceso en segundo plano redirigiendo la salida, y esta vez elimínalo usando con el comando kill el argumento del PID.

Yes > /dev/null &

Kill -15 1159

```
[centos@localhost ~]$ yes > /dev/null &
[1] 3613
[centos@localhost ~]$ kill -15 3613
[1]+  Terminated                  yes > /dev/null
[centos@localhost ~]$
```

9)Vuelve a lanzar el proceso en primer plano y suspéndelo con CTRL+Z.

Yes > /dev/null

Ctrl+z

```
[centos@localhost ~]$ yes > /dev/null
^Z
[1]+  Stopped                  yes > /dev/null
[centos@localhost ~]$
```

10) Ejecuta el comando jobs para ver el estado en que se encuentra.

Jobs

[6] Detenido

```
[centos@localhost ~]$ jobs
[1]+  Stopped                  yes > /dev/null
[centos@localhost ~]$
```

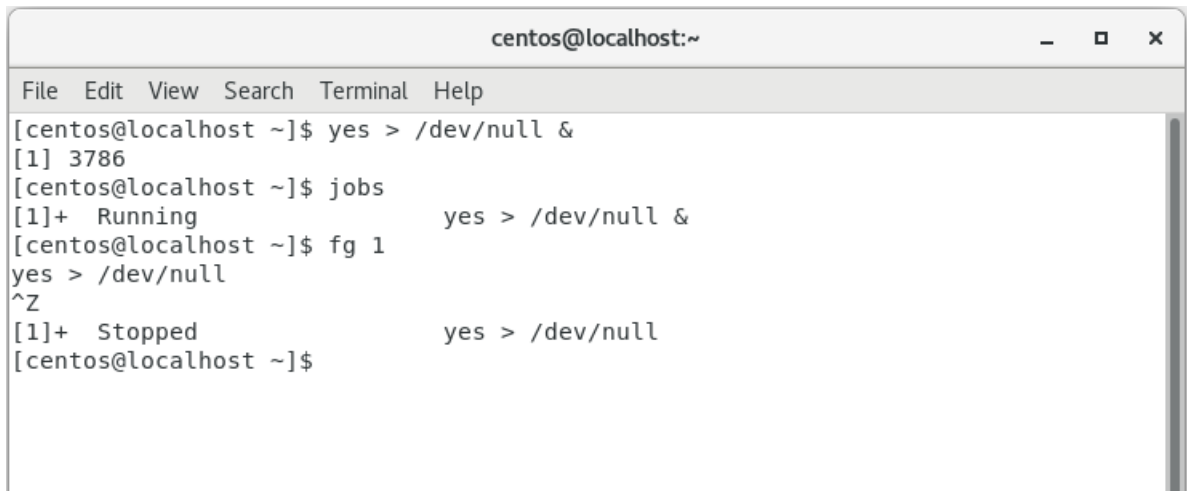
11) Relanza la tarea en primer plano con el comando fg.

Fg 6

```
centos@localhost:~
File Edit View Search Terminal Help
[centos@localhost ~]$ yes > /dev/null &
[1] 3786
[centos@localhost ~]$ jobs
[1]+  Running                  yes > /dev/null &
[centos@localhost ~]$ fg 1
yes > /dev/null
```

12) Suspende la ejecución del proceso con CTRL+Z.

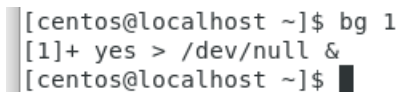
Ctrl+z

A terminal window titled 'centos@localhost:~' with a menu bar (File, Edit, View, Search, Terminal, Help). The terminal shows the following sequence of commands and output:

```
[centos@localhost ~]$ yes > /dev/null &
[1] 3786
[centos@localhost ~]$ jobs
[1]+  Running                  yes > /dev/null &
[centos@localhost ~]$ fg 1
yes > /dev/null
^Z
[1]+  Stopped                  yes > /dev/null
[centos@localhost ~]$
```

13) Relanza la tarea en segundo plano con el comando bg.

Bg 6

A terminal snippet showing the command 'bg 1' and its output:

```
[centos@localhost ~]$ bg 1
[1]+ yes > /dev/null &
[centos@localhost ~]$
```

14) Repite los cinco pasos anteriores, pero esta vez teniendo dos procesos. Puedes usar por ejemplo el comando sleep que realiza una pausa de un número determinado de segundos (ejemplo: sleep 1000).

Sleep 1000 > /dev/null

Ctrl+z

Sleep 1001 > /dev/null

Ctrl+z

Jobs

[7]- Detenido Sleep 1000 > /dev/null

[7]- Detenido Sleep 1001 > /dev/null


```

[centos@localhost ~]$ sleep 1000 > /dev/null
^Z
[2]+  Stopped                  sleep 1000 > /dev/null
[centos@localhost ~]$ sleep 1001 > /dev/null
^Z
[3]+  Stopped                  sleep 1001 > /dev/null
[centos@localhost ~]$ jobs
[1]  Running                  yes > /dev/null &
[2]-  Stopped                  sleep 1000 > /dev/null
[3]+  Stopped                  sleep 1001 > /dev/null
[centos@localhost ~]$ fg 2
sleep 1000 > /dev/null
^Z
[2]+  Stopped                  sleep 1000 > /dev/null
[centos@localhost ~]$ fg 3
sleep 1001 > /dev/null
^Z
[3]+  Stopped                  sleep 1001 > /dev/null

```

Ahora tenemos dos procesos sleep detenido y podemos relanzarlos en primer o segundo plano con las ordenes fg y bg.

```

[centos@localhost ~]$ bg 2
[2] sleep 1000 > /dev/null &
[centos@localhost ~]$ bg 3
[3]- sleep 1001 > /dev/null &
[centos@localhost ~]$

```

5) Filtrar los procesos por uso de CPU o memoria.

```

centos@localhost:~
File Edit View Search Terminal Help
top: unrecognized field name %cpu
[centos@localhost ~]$ top -o%CPU

top - 16:48:48 up 1:04, 5 users, load average: 1.28, 1.39, 1.36
Tasks: 203 total, 2 running, 198 sleeping, 3 stopped, 0 zombie
%Cpu(s): 98.7 us, 0.7 sy, 0.3 ni, 0.0 id, 0.0 wa, 0.0 hi, 0.3 si, 0.0 st
KiB Mem : 3861264 total, 1477048 free, 953912 used, 1430304 buff/cache
KiB Swap: 2097148 total, 2097148 free, 0 used. 2621356 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM    TIME+  COMMAND
 3733 centos    20   0 108056    356    280 R  94.0   0.0   23:14.13 yes
 2166 centos    20   0 3048816 232388 86896 S   3.6   6.0   1:00.34 gnome-shell
 1485 root       20   0 371580   95416 43440 S   1.3   2.5   0:23.45 X
 569 root       20   0 295376   5196  3968 S   0.7   0.1   0:05.09 vmtoolsd
 2758 centos   22   2 751876 33588 16764 S   0.7   0.9   0:48.03 gnome-terminal-
 2503 centos    20   0 610504 27528 18928 S   0.3   0.7   0:05.39 vmtoolsd
    1 root       20   0 128388   6972  4196 S   0.0   0.2   0:01.52 systemd
    2 root       20   0      0      0      0 S   0.0   0.0   0:00.00 kthreadd
    4 root       0 -20      0      0      0 S   0.0   0.0   0:00.00 kworker/0:0H
    6 root       20   0      0      0      0 S   0.0   0.0   0:00.13 ksoftirqd/0
    7 root       rt    0      0      0      0 S   0.0   0.0   0:00.01 migration/0
    8 root       20   0      0      0      0 S   0.0   0.0   0:00.00 rcu_bh
    9 root       20   0      0      0      0 S   0.0   0.0   0:00.70 rcu_sched
   10 root       0 -20      0      0      0 S   0.0   0.0   0:00.00 lru-add-drain
   11 root       rt    0      0      0      0 S   0.0   0.0   0:00.04 watchdog/0
   13 root       20   0      0      0      0 S   0.0   0.0   0:00.00 kdevtmpfs
   14 root       0 -20      0      0      0 S   0.0   0.0   0:00.00 netns
   15 root       20   0      0      0      0 S   0.0   0.0   0:00.00 khungtaskd
   16 root       0 -20      0      0      0 S   0.0   0.0   0:00.00 writeback
   17 root       0 -20      0      0      0 S   0.0   0.0   0:00.00 kintegrityd
   18 root       0 -20      0      0      0 S   0.0   0.0   0:00.00 bioset
   19 root       0 -20      0      0      0 S   0.0   0.0   0:00.00 bioset
   20 root       0 -20      0      0      0 S   0.0   0.0   0:00.00 bioset
   21 root       0 -20      0      0      0 S   0.0   0.0   0:00.00 kblockd
   22 root       0 -20      0      0      0 S   0.0   0.0   0:00.00 md

```

6) Filtrar los procesos que consumen mas procesador

```
[centos@localhost ~]$ ps aux --sort=-%cpu | head -n 11
```

USER	PID	%CPU	%MEM	VSZ	RSS	TTY	STAT	START	TIME	COMMAND
centos	3733	80.8	0.0	108056	356	pts/0	R+	16:22	14:08	yes
centos	3786	17.9	0.0	108056	356	pts/2	TN	16:22	3:01	yes
centos	2166	1.7	6.0	3048816	232328	?	Sl	15:44	0:57	/usr/bin/gnome-shell
centos	2758	1.4	0.8	751400	33584	?	SNL	15:44	0:47	/usr/libexec/gnome-terminal-server
root	1485	0.6	2.4	371580	95416	tty1	Ssl+	15:44	0:22	/usr/bin/X :0 -background none -noreset -audit 4 -verb
ose	-auth	/run/gdm/auth-for-gdm-NbKFPQ/database	-seat	seat0	-nolisten	tcp	vt1			
root	569	0.1	0.1	295376	5196	?	Ssl	15:43	0:04	/usr/bin/vmtoolsd
centos	2503	0.1	0.7	610504	27528	?	Sl	15:44	0:04	/usr/bin/vmtoolsd -n vmusr
centos	3451	0.1	0.0	162236	2316	pts/1	SN+	16:07	0:02	top
root	1	0.0	0.1	128388	6964	?	Ss	15:43	0:01	/usr/lib/systemd/systemd --switched-root --system --de

```
serialize 22
root      2  0.0  0.0      0      0 ?      S   15:43  0:00 [kthreadd]
[centos@localhost ~]$
```

7) Procesos que consumen mas memoria ram

```
[centos@localhost ~]$ ps aux --sort=-%mem | head -n 11
```

USER	PID	%CPU	%MEM	VSZ	RSS	TTY	STAT	START	TIME	COMMAND
centos	2166	1.7	6.0	3048816	232332	?	Sl	15:44	0:57	/usr/bin/gnome-shell
root	1485	0.6	2.4	371580	95416	tty1	Ssl+	15:44	0:22	/usr/bin/X :0 -background none -noreset -audit 4 -verb
ose	-auth	/run/gdm/auth-for-gdm-NbKFPQ/database	-seat	seat0	-nolisten	tcp	vt1			
centos	2519	0.0	1.4	1104880	55968	?	Sl	15:44	0:00	/usr/bin/gnome-software --gapplication-service
centos	2758	1.4	0.8	751400	33584	?	RNL	15:44	0:47	/usr/libexec/gnome-terminal-server
centos	2453	0.0	0.8	1023760	33008	?	Sl	15:44	0:00	nautilus-desktop --force
root	685	0.0	0.7	358868	29592	?	Ssl	15:43	0:00	/usr/bin/python2 -Es /usr/sbin/firewalld --nofork --no
pid										
centos	2248	0.0	0.7	902784	29016	?	Sl	15:44	0:00	/usr/libexec/goa-daemon
centos	2503	0.1	0.7	610504	27528	?	Sl	15:44	0:05	/usr/bin/vmtoolsd -n vmusr
root	1068	0.0	0.5	1006580	20572	?	Ssl	15:44	0:00	/usr/sbin/libvirtd
centos	2434	0.0	0.5	1077676	20008	?	Sl	15:44	0:00	/usr/libexec/evolution-calendar-factory-subprocess --f

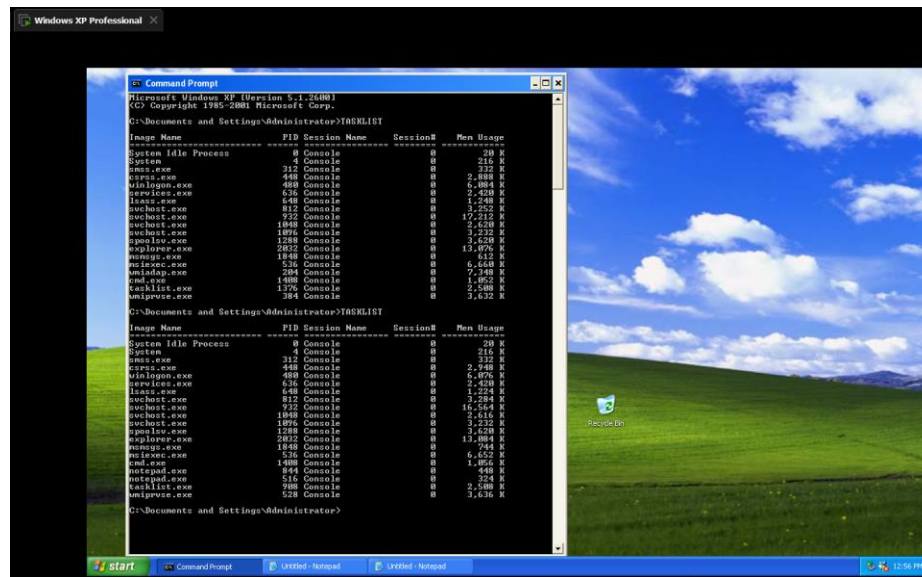
```
actory all --bus-name org.gnome.evolution.dataserver.Subprocess.Backend.Calendarx2377x2 --own-path /org/gnome/evolution/
dataserver/Subprocess/Backend/Calendar/2377/2
[centos@localhost ~]$
```

8) Mostrar los procesos en forma jerárquica

```
[centos@localhost ~]$ ps axjf
```

PPID	PID	PGID	SID	TTY	TPGID	STAT	UID	TIME	COMMAND
0	2	0	0	?	-1	S	0	0:00	[kthreadd]
2	4	0	0	?	-1	S<	0	0:00	_ [kworker/0:0H]
2	6	0	0	?	-1	S	0	0:00	_ [ksoftirqd/0]
2	7	0	0	?	-1	S	0	0:00	_ [migration/0]
2	8	0	0	?	-1	S	0	0:00	_ [rcu_bh]
2	9	0	0	?	-1	R	0	0:00	_ [rcu_sched]
2	10	0	0	?	-1	S<	0	0:00	_ [lru-add-drain]
2	11	0	0	?	-1	S	0	0:00	_ [watchdog/0]
2	13	0	0	?	-1	S	0	0:00	_ [kdevtmpfs]
2	14	0	0	?	-1	S<	0	0:00	_ [netns]
2	15	0	0	?	-1	S	0	0:00	_ [khungtaskd]
2	16	0	0	?	-1	S<	0	0:00	_ [writeback]
2	17	0	0	?	-1	S<	0	0:00	_ [kintegrityd]
2	18	0	0	?	-1	S<	0	0:00	_ [bioset]
2	19	0	0	?	-1	S<	0	0:00	_ [bioset]
2	20	0	0	?	-1	S<	0	0:00	_ [bioset]
2	21	0	0	?	-1	S<	0	0:00	_ [kblockd]
2	22	0	0	?	-1	S<	0	0:00	_ [md]
2	23	0	0	?	-1	S<	0	0:00	_ [edac-poller]
2	24	0	0	?	-1	S<	0	0:00	_ [watchdogd]
2	30	0	0	?	-1	S	0	0:00	_ [kswapd0]
2	31	0	0	?	-1	SN	0	0:00	_ [ksmd]
2	32	0	0	?	-1	SN	0	0:00	_ [khugepaged]
2	33	0	0	?	-1	S<	0	0:00	_ [crypto]
2	41	0	0	?	-1	S<	0	0:00	_ [kthrotld]
2	43	0	0	?	-1	S<	0	0:00	_ [kmpath_rdacd]
2	44	0	0	?	-1	S<	0	0:00	_ [kaluad]
2	45	0	0	?	-1	S<	0	0:00	_ [kpsmoused]
2	47	0	0	?	-1	S<	0	0:00	_ [ipv6_addrconf]
2	60	0	0	?	-1	S<	0	0:00	_ [deferwq]
2	96	0	0	?	-1	S	0	0:00	_ [kauditd]

9) Ejecute el comando TASKLIST en su computadora con Windows y explique para que sirve esta instrucción.

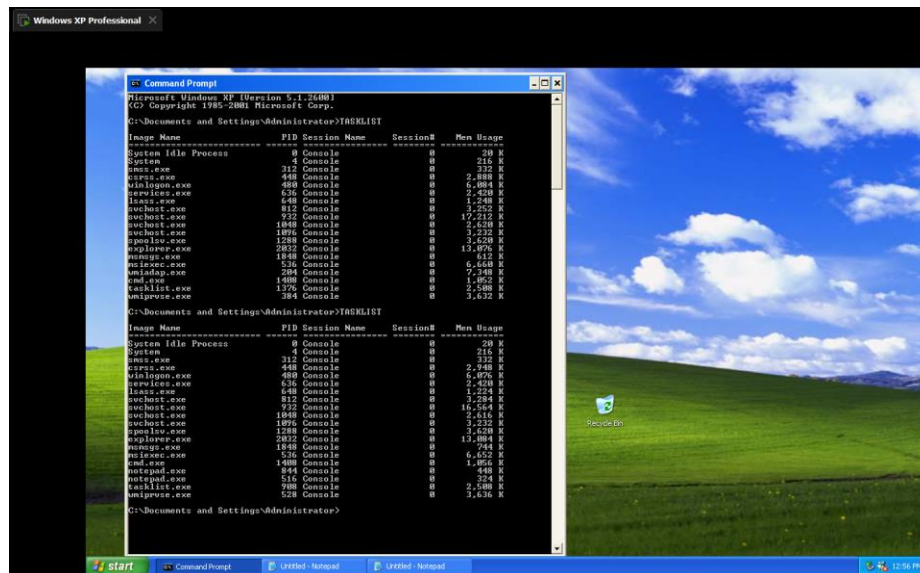


The screenshot shows a Windows XP Professional desktop with a green field and blue sky wallpaper. A Command Prompt window is open, displaying the output of the TASKLIST command. The output is a table with columns: Image Name, PID, Session Name, SessionID, and Mem Usage. The table lists various system and user processes, including System Idle Process, System, smss.exe, csrss.exe, winlogon.exe, services.exe, lsass.exe, smss.exe, csrss.exe, explorer.exe, notepad.exe, cmd.exe, tasklist.exe, and vmtoolsd.exe. The Command Prompt window title is 'Command Prompt' and the background window title is 'Windows XP Professional'.

Image Name	PID	Session Name	SessionID	Mem Usage
System Idle Process	0	Console	0	28 K
System	4	Console	0	216 K
smss.exe	312	Console	0	312 K
csrss.exe	448	Console	0	2,348 K
winlogon.exe	488	Console	0	5,084 K
services.exe	636	Console	0	2,420 K
lsass.exe	648	Console	0	1,224 K
smss.exe	812	Console	0	3,284 K
csrss.exe	924	Console	0	16,544 K
smss.exe	1048	Console	0	2,616 K
csrss.exe	1096	Console	0	3,224 K
explorer.exe	1288	Console	0	3,620 K
notepad.exe	2032	Console	0	13,884 K
cmd.exe	1848	Console	0	744 K
tasklist.exe	528	Console	0	6,652 K
vmtoolsd.exe	1488	Console	0	1,064 K
notepad.exe	844	Console	0	448 K
tasklist.exe	516	Console	0	224 K
vmtoolsd.exe	788	Console	0	2,388 K
vmtoolsd.exe	528	Console	0	3,636 K

En esta ocasión use un maquina virtual con el sistema operativo de Windows la versión XP. El comando TASKLIST como se pudo observar en la practica es una herramienta de línea de comandos que se utiliza para mostrar una lista como su nombre nos indica, de todos los procesos que se están ejecutando en el sistema, esta muestra información detallada sobre los procesos, como el nombre del proceso, el PID, el uso de CPU, el uso de memoria y la hora de inicio del proceso.

- 10) Para que sirve el comando TASKILL , ejecute 2 procesos en este caso abra 2 notepad, liste los procesos en Windows y ejecute el comando TASKILL en la consola de comando.



Como se puede observar el PID de nuestro proceso en este caso el Notepad.exe es el 844 y el 516 ya que se menciona que hay que abrir dos notepads.



El comando TASKKILL se utiliza como se vio en la práctica, para terminar procesos o aplicaciones por su nombre de imagen también por su ID de procesos en este caso el PID o por su nombre de ventana, ofrece varias opciones para controlar la forma en la que se termina el proceso como / F (fuerza bruta) termina el proceso sin avisar al usuario.