Universidad de Antioquia

Facultad de Ingeniería

Sistemas Operativos y Laboratorio

Tarea de Programación Laboratorio 2: Desarrollo del programa psinfo

# 1. Introducción

En una empresa de desarrollo de software de sistema se está realizando un proyecto de personalización de un sistema operativo Linux Ubuntu. Los diseñadores del sistema piensan incluir nuevos comandos en el intérprete de línea de comandos incluido en la distribución. A usted se le ha asignado la tarea de programar un nuevo comando el cual permite obtener información importante acerca de un determinado **proceso** del sistema operativo. El comando se llamará **psinfo**.

# 2. Contextualización: Información de procesos en Linux

## 2.1. Sobre los procesos

Existe un dicho dado por los genio de Free Electrons y es “Todo en Unix es un archivo y todo lo que no es archivo en Unix es un proceso”[[1]](#footnote-1). Recordemos que un proceso es una instancia de un programa en ejecución la cual tiene una serie de datos asociados como: Archivos abiertos, memoria asignada, identificador de proceso, padre, prioridad, estado, entre otros. Tanto en Ubuntu como en Linux se puede obtener algo de información relacionada con los procesos en ejecución al ejecutar el administrador de tareas. Para llamarlo en Ubuntu, en el menú se puede digitar **System Monitor** tal y como se muestra en la figura 1.

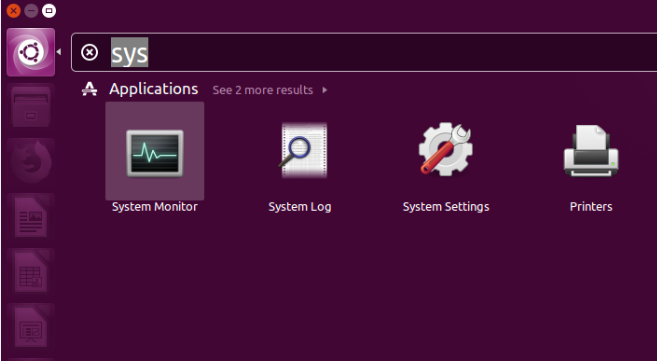


Figura 1. Inicio del Monitor del Sistema (System Monitor)

Una vez que se abre el monitor del sistema, se pueden ver algunos de los procesos en ejecución e información relacionada (Figura 2), por ejemplo su identificador de proceso (PID).

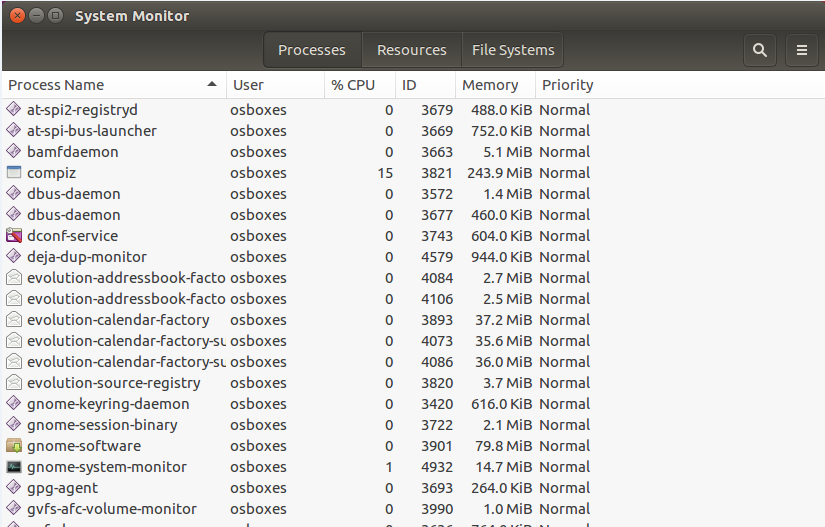


Figura 2. System Monitor

En la siguiente sección se verán algunos comandos con el fin de analizar cómo se obtiene esta misma información desde consola.

## 2.2. Procesos y consola

Desde la consola de linux es posible obtener información relacionada con los procesos. A continuación se presenta una lista de los comandos más populares utilizado para obtener información de procesos. Si quiere conocer más a fondo el uso de los comandos para obtener información de procesos se sugiere el texto guia del curso ([www.google.com](http://www.google.com)) y las referencias al final del documento. En la siguiente tabla se da una breve descripción del comando ps:

* **Comando ps**

|  |
| --- |
| ps <opciones> |

Lista los procesos que se están ejecutando en el sistema, si no se agrega ningún parámetro opcional ps mostrará los procesos del usuario que se encuentra logueado. Las opciones adicionales de este comando se pueden consultar en el manual o en las referencias al final del documento. Por ejemplo el comando:

|  |
| --- |
| ps ­ux |

Lista todos los procesos correspondientes al usuario actual. Mientras el comando:

|  |
| --- |
| ps ­aux |

Lista todos los procesos que se están ejecutando en el sistema.

* **Comando top**

|  |
| --- |
| top |

Despliega los procesos más importantes ordenados de acuerdo al uso de porcentaje de CPU. Debido a que en este comando la información se despliega (más o menos) en tiempo real, la consola se bloquea de modo que para salir de este se presiona la tecla **q**.

* **Comando jobs**

|  |
| --- |
| jobs |

Retorna la lista de procesos en background en el mismo shell. Lanzar procesos en background es de utilidad si no se quiere bloquear la consola. Para hacer que un proceso se ejecute en background, se debe agregar & al final de la línea. Así:

|  |
| --- |
| comando & |

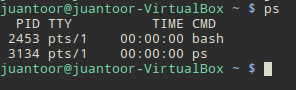
Por ejemplo, si hubiésemos deseado correr gedit en background, el comando sería:

|  |
| --- |
| gedit & |

## 2.3. Practicando los comandos anteriores

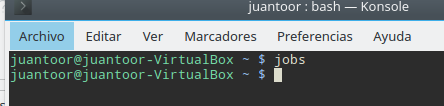
A continuación se muestra paso a paso la ejecución de los comandos anteriormente mencionados en una máquina determinada con el fin de facilitar su comprensión. Pese a que se muestran capturas y parte de las salidas se recomienda que los ejecute en su máquina y compare los resultados, su comprensión puede ser de utilidad para el desarrollo de la práctica.

1. Abra una terminal de linux y ejecute el comando ps. ¿Qué observa?



|  |
| --- |
| $ ps  PID TTY TIME CMD  5151 pts/4 00:00:00 bash  5204 pts/4 00:00:00 ps |

1. Ejecute el comando jobs. ¿Qué observa y por qué?



1. Ahora llame el editor de texto de su preferencia (en este ejemplo se llamó gedit) en background. Si usted observa la salida verá un número ¿Con qué se relaciona este?



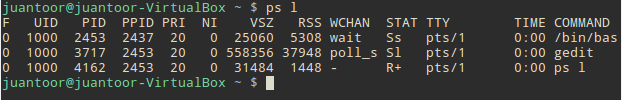
|  |
| --- |
| $ gedit &  [1] 5205 |

1. Ejecute nuevamente el comando jobs. ¿Cambia la salida?



|  |
| --- |
| $ jobs  [1]+ Running gedit & |

1. Ahora ejecute el comando ps con la opción l



|  |
| --- |
| $ ps l  F UID PID PPID PRI NI VSZ RSS WCHAN STAT TTY TIME COMMAND  0 1000 5151 5144 20 0 22636 5240 wait Ss pts/4 0:00 bash  0 1000 5205 5151 20 0 655856 36848 poll\_s Sl pts/4 0:07 gedit  0 1000 6339 5151 20 0 28916 1620 - R+ pts/4 0:00 ps l |

A continuación se describen algunos de los ítems (los cuales se muestran en la salida anterior) asociados con el proceso :

* **PID**: Process id
* **VSZ**: Virtual process size (code + data + stack)
* **RSS**: Process resident
* **size**: number of KB currently in RAM
* **TTY**: Terminal STAT:
* **Status**: R (Runnable), S (Sleep), W (paging), Z (Zombie)...

¿Que significado tiene dentro de lo que se ha visto en la teoría cada una de estas salidas? Llene una tabla con la información con estos datos y que le fue arrojada en su caso particular.

F PROCESS FLAGS

1 bifurcado pero no ejecutado.

4 tiene privilegios de root.

UID ID de usuario.

PID ID del proceso padre

PPID ID del proceso padre.

PGID ID de grupo de un proceso.

PRI Prioridad del proceso.

NI valor de bondad, más elevado menor prioridad.

VSZ Tamaño de la memoria virtual del proceso en Kb.

RSS Tamaño de la memoria física usada en Kb.

WCHAN para los procesos que esperan o dormidos, enumera el evento que espera.

STAT Estado del proceso:

R Ejecutable.

D Interrumpió.

S Suspendido.

s Es el proceso líder de la sesión.

T Detenido.

Z Zombie.

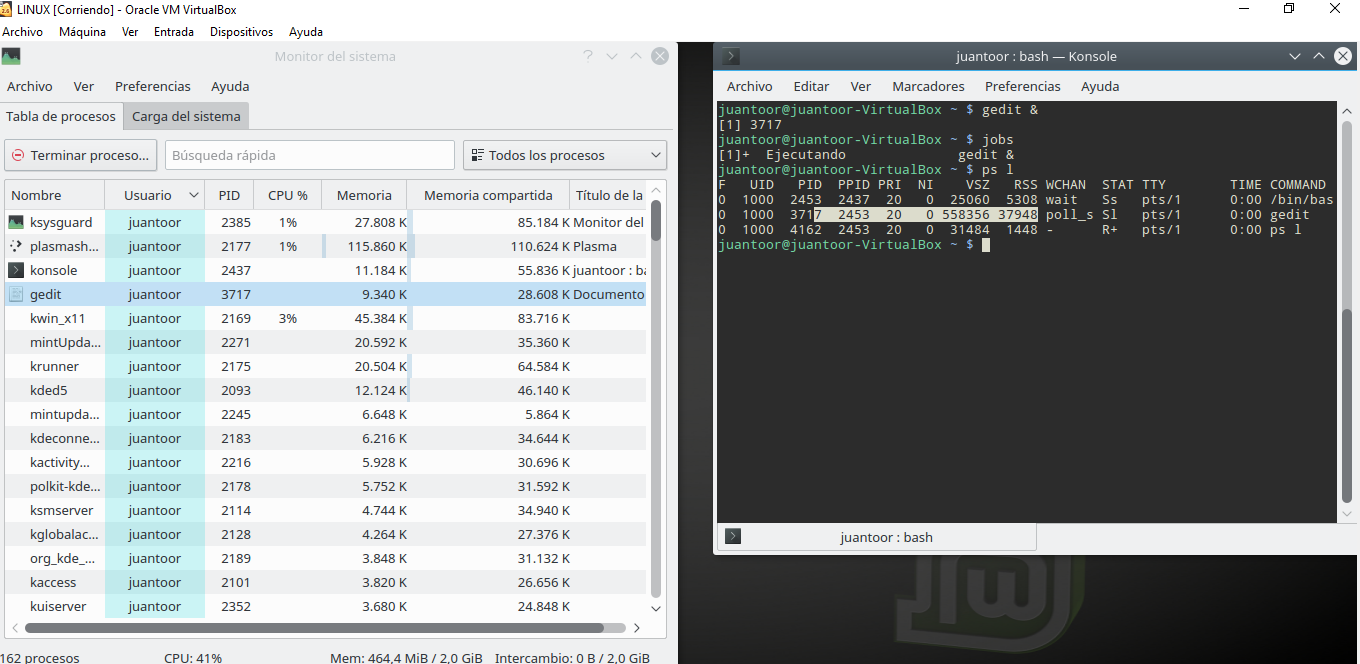
N Tiene una prioridad menor que lo normal.

< Tiene una prioridad mayor que lo normal.

TTY nombre de la terminal a la que esta asociado al proceso.

TIME tiempo que lleva en ejecución.

1. Desde el monitor del sistema observe la información asociada al proceso del gedit (ver Figura 3). Compare los resultados con la salida del comando ejecutado en el paso 5



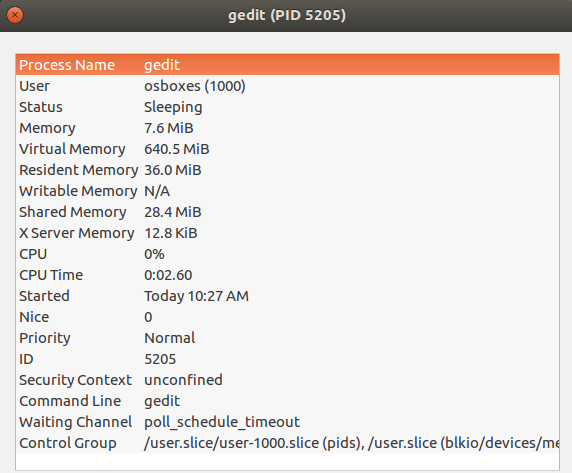
1. 

Figura 3. Información del proceso gedit

1. Cierre el programa gedit y ejecute nuevamente el comando ps. ¿Por qué la salida?

|  |
| --- |
| $ ps  PID TTY TIME CMD  5151 pts/4 00:00:00 bash  6559 pts/4 00:00:00 ps  [1]+ Killed gedit |

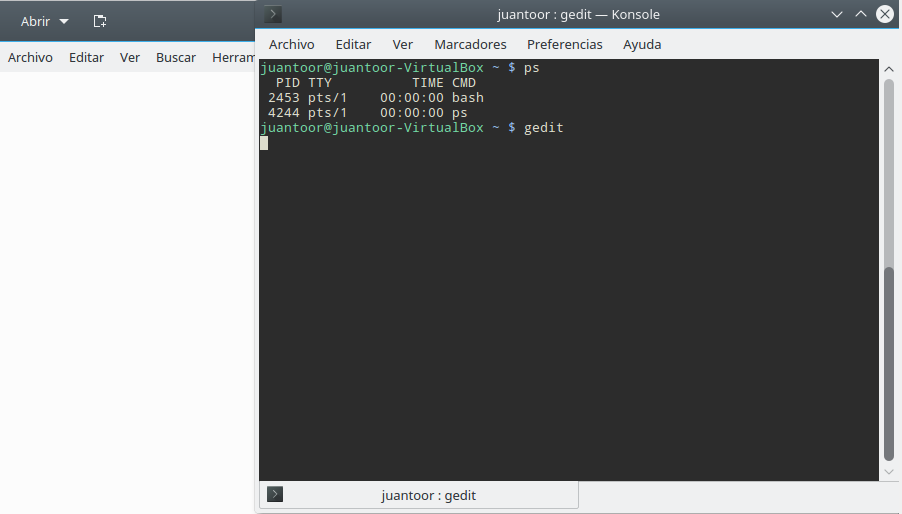
## 2.4. Sobre el sistema de archivos procfs:

El sistema de archivos /proc se utiliza para ofrecer información relacionada con el sistema. [En este enlace](http://man7.org/linux/man-pages/man5/proc.5.html) pueden consultar la información detallada contenida en esta carpeta. Con el fin de facilitar la implementación de lo que se pide en el laboratorio puede ser de utilidad usar algunos comandos de consola, para ello trate de seguir las siguientes actividades orientado por el tutor:

**Actividad 1**:

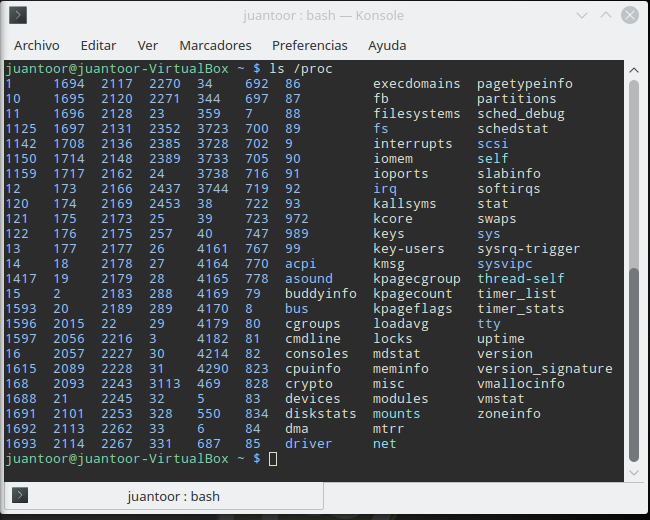
Suponga que usted tiene una consola abierta y ejecute el comando ps. Luego ejecute el comando gedit para llamar el editor de textos:

|  |
| --- |
| $ gedit &  [1] 4858 |



**Actividad 2**:

Ahora explore el contenido del directorio /proc, esto lo puede hacer con el comando ls, tal y como se muestra a continuación en pantalla. Consulte y describa brevemente qué información aparece allí.



|  |
| --- |
| $ ls /proc  1 173 2493 2740 2962 3283 84 cmdline misc  10 176 25 2741 299 33 85 consoles modules  102 18 250 2749 2991 333 856 cpuinfo mounts  1073 183 2507 2755 2993 3334 86  ... |

Este directorio especial contiene todos los detalles de tu sistema Linux, incluyendo el kernel, procesos y parámetros de configuración.

**Preguntas**:

1. ¿Qué significan todos esos numeros enteros? (Consulte el enlace de arriba).
2. ¿Estos números son archivos o directorios?
3. El número entero 1 ¿con qué está asociado?

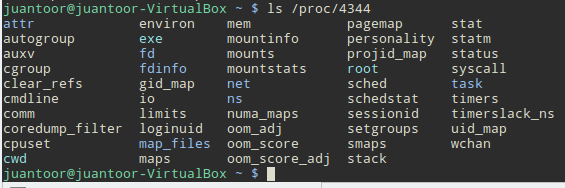
Los directorios con números (más sobre esto en un momento) corresponden a cada proceso en ejecución; un autoenlace simbólico apunta al proceso actual. Algunos archivos virtuales proveen información sobre el hardware, tal como /proc/cpuinfo y /proc/interrupts. Otros proporcionan información relacionada a los archivos, tales como /proc/filesystems o /proc/partitions. Los archivos situados bajo /proc/sys son relacionados a la configuración de parámetros del kernel, como ya veremos.

/proc/1

Un directorio con información acerca del proceso número 1. Cada proceso tiene un directorio debajo de /proc cuyo nombre es el número de identificación del proceso (PID).

**Actividad 3**:

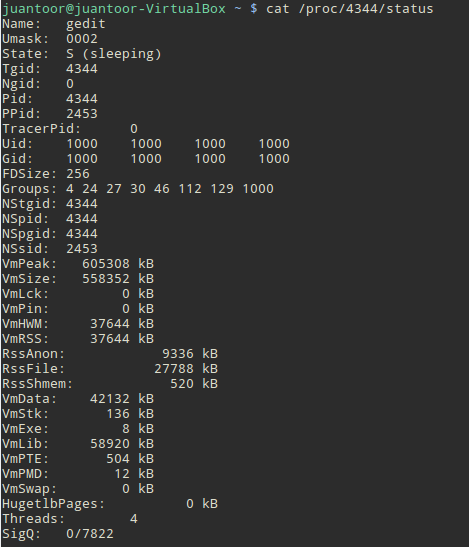
Ahora explore el contenido del número asociado al gedit.

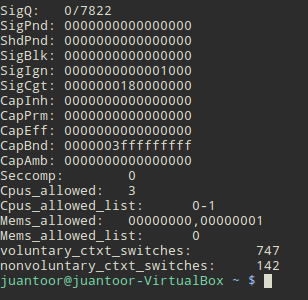


|  |
| --- |
| $ ls /proc/**4858**  attr cwd map\_files oom\_adj sessionid timers  autogroup environ maps oom\_score setgroups timerslack\_ns  auxv exe mem oom\_score\_adj smaps uid\_map  cgroup fd mountinfo pagemap stack wchan  clear\_refs fdinfo mounts personality stat  cmdline gid\_map mountstats projid\_map statm  comm io net root status  coredump\_filter limits ns sched syscall  cpuset loginuid numa\_maps schedstat task |

**Preguntas**:

1. ¿Qué información se obtiene de los archivos stat y status anteriormente resaltados?
2. Teniendo en cuenta lo anterior llene la siguiente tabla para el proceso asociado al editor de textos. Llene la siguiente tabla para su caso particular.





|  |  |
| --- | --- |
| **Información** | **Valor** |
| Name |  |
| State |  |
| FDSize |  |
| VmPeak |  |
| VmSize |  |
| VmLck |  |
| voluntary\_ctxt\_switches |  |
| nonvoluntary\_ctxt\_switches |  |

# 3. Descripción del programa a desarrollar psinfo

Para el desarrollo de este programa se ha definido una metodología por etapas donde cada etapa debe proporcionar un producto funcional completo. psinfo es una aplicación hecha en C cuyo objetivo es implementar de manera más simple comandos como los que se ejecutaron anteriormente mediante la lectura de archivos del sistema, en este caso los contenidos en el directorio /proc.

Se inicia con el desarrollo de una implementación muy simple de la función aumentando su funcionalidad etapa por etapa.

## 3.1. Etapa 1

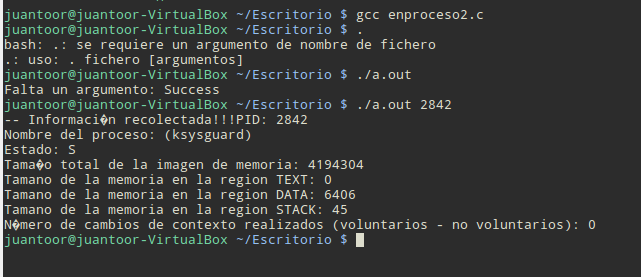
Se requiere implementar el programa básico **psinfo**, el cual recibe como argumento de entrada un identificador de proceso (pid) y retorna en pantalla una información de estado del proceso. La información de estado solicitada es la siguiente:

1. Nombre del proceso.
2. Estado del proceso.
3. Tamaño total de la imagen de memoria.
   1. Tamaño de la sección de memoria TEXT.
   2. Tamaño de la sección de memoria DATA.
   3. Tamaño de la sección de memoria STACK.
4. Número de cambios de contexto realizados:
   1. Voluntarios.
   2. No voluntarios.

La información debe tomarse del pseudo sistema de archivos de procesos del sistema operativo linux *procfs* (ubicado en la ruta /proc). [En este enlace](http://man7.org/linux/man-pages/man5/proc.5.html) pueden consultar la información detallada contenida en esta carpeta. Se sugiere revisar especialmente el archivo **/proc/[pid]/status**, pues es el que contiene la información solicitada.

Ejemplo de uso:

|  |
| --- |
| $ ./psinfo 10898  Nombre del proceso: gedit  Estado: S (sleeping)  Tamaño total de la imagen de memoria: 715000 KB  Tamaño de la memoria en la región TEXT: 10000KB  Tamaño de la memoria en la región DATA: 24200KB  Tamaño de la memoria en la región STACK: 21000KB  Número de cambios de contexto realizados (voluntarios - no voluntarios): 17536 - 189 |



## 3.2. Etapa 2

Implementar en el comando la opción **lista** (-l) en la cual el usuario pueda pasar una lista de identificadores de procesos. Para cada uno de los procesos en la lista, el programa debe obtener la información de estado, almacenarla en una estructura de memoria temporal (usando estructuras de C y memoria dinámica) y reportar los valores almacenados en pantalla.

Ejemplo de uso:

|  |
| --- |
| $ psinfo -l 10898 1342 2341  -- Información recolectada!!!  Pid: 10898  Nombre del proceso: gedit  …  Pid: 1342  Nombre del proceso: chrome  …  Pid: 2341  Nombre del proceso: bash  … |

## 3.3. Etapa 3

Implementar una opción de reporte (-r) en la cual el proceso realice la funcionalidad solicitada en la etapa 2, pero generando un archivo de salida con la información de estado de los procesos solicitados. El archivo de salida debe llamarse psinfo-report-[pids].info.

Ejemplo de uso:

|  |
| --- |
| $ psinfo -r 10898 1342  Archivo de salida generado: psinfo-report-10898-1342.info |

## 3.4. Etapa 4

Implemente las validaciones necesarias para identificar un uso incorrecto del programa. Por ejemplo, detectando parámetros no válidos o solicitud de información de procesos inexistentes. Si el programa detecta un mal uso debe imprimir un mensaje informando acerca del error detectado, luego debe imprimir un mensaje mostrando un resumen de los parámetros válidos y debe finalizar el programa reportando un código de ejecución errónea al *bash*. El código de error puede ser definido por usted mismo y debe ser diferente al valor correspondiente a la ejecución normal.

# 4. Consideraciones generales

* Desarrolle el programa por etapas conservando siempre la última versión funcional de su implementación (se sugiere hacer uso de repositorios para este fin: git o svn).
* Realice su implementación teniendo en cuenta todas las buenas prácticas de programación explicadas en clase de laboratorio.
* Realice una implementación orientada a funciones. Tenga en cuenta que el código que usted desarrolle debe ser funcional y, en lo posible, eficiente.
* Ante cualquier duda consulte con el equipo de profesores del laboratorio en los medios dispuestos para tal fin (foros piazza o moodle).

# 5. Enlaces

1. 20 comandos para administrar y gestionar facilmente los procesos en Linux (Accedido el 6 de marzo de 2018: <https://openwebinars.net/blog/20-comandos-para-administrar-y-gestionar--facilmente-los-procesos-linux/>)
2. Obtener información básica sobre procesos del sistema Linux en C y C++ (parte 3) (Accedido el 6 de marzo de 2018: <https://poesiabinaria.net/2017/07/obtener-informacion-basica-procesos-del-sistema-linux-c-c-parte-3/>)

1. Ver diapositiva 65 del documento [The Unix and GNU/Linux command line](https://bootlin.com/doc/legacy/command-line/unix_linux_introduction.pdf) [↑](#footnote-ref-1)