



INSTITUTO POLITÉCNICO NACIONAL.  
ESCUELA SUPERIOR DE CÓMPUTO.



# SISTEMAS OPERATIVOS.

## PRÁCTICA 2

### ENTORNO DE LINUX

#### Integrantes del equipo:

- Chavarria Vázquez Luis Enrique.
- Juárez Espinoza Ulises.
- Machorro Vences Ricardo Alberto.
- Pastrana Torres Víctor Norberto.



# Índice de contenido.

<b>Glosario de términos</b> .....	1
<b>Máquina virtual</b> .....	1
<b>Contenido</b> .....	2
<b>Escritorios GNOME y KDE</b> .....	2
<b>Entornos Command Line Interface y Graphical User Interface</b> .....	3
<b>Terminal de Linux</b> .....	4
<b>Tipos de usuario en Linux</b> .....	5
<b>Direccionamiento relativo y absoluto.</b> .....	6
<b>Redireccionamiento.</b> .....	8
FILE DESCRIPTORS – FD (Descriptores de archivos) .....	8
REDIRECTION OPERATORS (Operadores de redireccionamiento).....	8
<b>Clasificación de los comandos en Linux.</b> .....	9
Información del sistema .....	9
Apagar (Reiniciar Sistema o Cerrar Sesión) .....	10
Archivos y Directorios .....	10
Encontrar archivos.....	11
Montando un sistema de ficheros.....	12
Espacio de Disco .....	12
Usuarios y Grupos.....	13
Permisos en Ficheros (Usa "+" para colocar permisos y "-" para eliminar) .....	13
Atributos especiales en ficheros (Usa "+" para colocar permisos y "-" para eliminar).....	14
Archivos y Ficheros comprimidos.....	15
Paquetes RPM (Red Hat, Fedora y similares).....	15
Actualizador de paquetes YUM (Red Hat, Fedora y similares).....	16
Paquetes Deb (Debian, Ubuntu y derivados). ....	17
Actualizador de paquetes APT (Debian, Ubuntu y derivados). .....	17
Ver el contenido de un fichero. ....	18
Manipulación de texto.....	18
Establecer carácter y conversión de ficheros. ....	19
Análisis del sistema de ficheros.....	19
Formatear un sistema de ficheros .....	20
Trabajo con la SWAP.....	20
Salvas (Backup). .....	20

<b>CD-ROM.....</b>	<b>21</b>
Trabajo con la RED ( LAN y Wi-Fi).....	21
Tablas IP (CORTAFUEGOS) .....	22
Monitoreando y depurando.....	23
Comandos adicionales.....	24
<b>Variables de entorno.....</b>	<b>24</b>
Variables globales .....	25
Variables locales. ....	25
Declarar variables locales.....	25
Declarar variables globales (variables de exportación).....	25
<b>Desarrollo (Códigos y ventanas de ejecución).....</b>	<b>27</b>
<b>Comando cal .....</b>	<b>29</b>
<b>Comando clear antes de ejecutarse .....</b>	<b>29</b>
<b>Comando clear después de ejecutarse.....</b>	<b>30</b>
<b>Comando apt .....</b>	<b>30</b>
<b>Comando rm.....</b>	<b>30</b>
<b>Comando Date.....</b>	<b>31</b>
<b>Comando ifconfig.....</b>	<b>31</b>
<b>Comando exit antes de ejecutarse.....</b>	<b>31</b>
<b>Comando exit después de ejecutarse.....</b>	<b>32</b>
<b>Comando mv.....</b>	<b>32</b>
<b>Comando echo .....</b>	<b>33</b>
<b>Comando df .....</b>	<b>33</b>
<b>Comando ps .....</b>	<b>34</b>
<b>Comando more .....</b>	<b>34</b>
<b>Comando time .....</b>	<b>35</b>
<b>Comando du.....</b>	<b>35</b>
<b>Comando ps –fea .....</b>	<b>36</b>
<b>Comando less antes de ejecutarse .....</b>	<b>36</b>
<b>Comando less después de ejecutarse .....</b>	<b>36</b>
<b>Función uname .....</b>	<b>37</b>
<b>Comando pstree.....</b>	<b>37</b>
<b>Comando man .....</b>	<b>38</b>
<b>Comando mkdir .....</b>	<b>38</b>
<b>Comando w .....</b>	<b>38</b>

<b>Comando Kill -9</b> .....	39
<b>Función cat</b> .....	39
<b>Comando pico antes de compilarse</b> .....	39
<b>Comando pico después de compilarse</b> .....	40
<b>Comando who</b> .....	40
<b>Comando fg</b> .....	40
<b>Comando nano antes de ejecutarse</b> .....	40
<b>Comando nano después de ejecutarse</b> .....	40
<b>Comando pwd</b> .....	40
<b>Comando cd</b> .....	40
<b>Comando vi antes de ejecutarse</b> .....	41
<b>Comando vi luego de ejecutarse</b> .....	41
<b>Comando wc</b> .....	41
<b>Comando ls</b> .....	41
<b>Comando apt-get</b> .....	42
<b>Comando sudo</b> .....	42
<b>Direccionamiento absoluto y relativo (3 ejemplos por tipo explicados de forma conjunta)</b> . .....	43
Ejemplo 1 (ruta absoluta y relativa) .....	43
Ejemplo 2 (ruta absoluta y relativa) .....	43
Ejemplo 3 (ruta absoluta y relativa) .....	44
<b>Borrado de archivos “rm y comodines *?”</b> .....	45
Ejemplo 1 Borrado con “rm” de un archivo dentro de 3 carpetas. ....	45
Ejemplo 2 Borrado con “rm” con el uso del comodín *. ....	46
Ejemplo 3 Borrado con “rm” con el uso del comodín ?....	46
Ejemplo 4 Borrado con “rm”.....	47
<b>Redireccionamiento utilizando ‘&gt;’ (crea archivo, sobrescribe si existe)</b> . .....	47
Ejemplo 1 .....	47
Ejemplo 2 .....	48
<b>Redireccionamiento usando ‘&gt;&gt;’ (crea si no existe, agrega al archivo existente)</b> . .....	49
Ejemplo 1 .....	49
Ejemplo 2 .....	50
<b>Instalar en línea de comandos del paquete gimp</b> .....	50
<b>Instalar slack por el entorno gráfico.</b> .....	51
<b>Jerarquía de directorios</b> .....	52
Ejemplo 1 .....	52

Ejemplo 2 .....	53
Ejemplo 3 .....	54
Ejemplo 4 .....	55
Ejemplo5 .....	56
<b>Secuencia de pasos para crear, verificar y eliminar un usuario .....</b>	<b>57</b>
Método 1 para crear un nuevo usuario.....	57
Método 2 para crear un nuevo usuario.....	58
<b>Verificación de la existencia de usuarios .....</b>	<b>59</b>
Método 1 forma gráfica .....	59
Método 2 Terminal.....	59
Método 3 Terminal.....	60
Cambiar de usuario.....	61
Eliminar un usuario. ....	61
<b>Compilar Hola Mundo en código para nano.....</b>	<b>62</b>
Creación Hola Mundo .....	62
Código Hola Mundo editado en nano .....	62
Código Hola Mundo compilación nano.....	62
<b>Compilar Hola Mundo en código para vi.....</b>	<b>63</b>
Creación Hola Mundo .....	63
Edición Hola Mundo en vi .....	63
Ejecución h ola mundo en vi .....	63
<b>Compilar Hola Mundo en código para pico.....</b>	<b>63</b>
Instalación de pico (Pine Composer). .....	63
Revisión de los comandos de instalación.....	63
Actualización de apt .....	64
Buscando soporte para pico en sitios oficiales.....	64
<b>Compilar Hola Mundo en código para GNU Emacs. (Para compensar pico) .....</b>	<b>66</b>
Instalación de Emacs.....	66
Escritura del programa “Hola mundo”. .....	66
Compilación y ejecución del programa dentro del editor. .....	66
<b>Código de programa1a.c.....</b>	<b>67</b>
<b>Resultado compilación Programa1a.c.....</b>	<b>67</b>
<b>Compilación y pantalla de programa1a.c.....</b>	<b>67</b>
<b>Código de programa2a.c.....</b>	<b>67</b>
<b>Compilación y pantalla de programa2a.c.....</b>	<b>68</b>

<b>Código de programa3a.c .....</b>	68
<b>Compilación y pantalla de programa3a.c .....</b>	70
<b>Conclusiones. ....</b>	71
<b>Chavarría Vázquez Luis Enrique.</b>	71
<b>Juárez Espinoza Ulises.</b>	73
<b>Machorro Vences Ricardo Alberto.</b>	75
<b>Pastrana Torres Victor Norberto.</b>	76
<b>Bibliografía .....</b>	78

# Índice de figuras

Ilustración 1. Ruta relativa .....	6
Ilustración 2. Ruta absoluta .....	7
Ilustración 3. Declarando una variable global .....	26
Ilustración 4. Declarando una variable global permanente .....	26
Ilustración 5. Comando cal .....	29
Ilustración 6. Comando clear antes de ejecutarse .....	29
Ilustración 7. Comando clear después de ejecutarse .....	30
Ilustración 8. Comando apt .....	30
Ilustración 9. Comando rm .....	30
Ilustración 10. Comando date .....	31
Ilustración 11. Comando ifconfig .....	31
Ilustración 12. Comando exit antes de ejecutarse .....	31
Ilustración 13. Comando exit después de ejecutarse .....	32
Ilustración 14. Comando mv .....	32
Ilustración 15. Comando echo .....	33
Ilustración 16. Comando df .....	33
Ilustración 17. Comando ps .....	34
Ilustración 18. Comando more .....	34
Ilustración 19. Comando time .....	35
Ilustración 20. Comando du .....	35
Ilustración 21. Comando ps -fea .....	36
Ilustración 22. Comando less antes de ejecutarse .....	36
Ilustración 23. Comando less después de ejecutarse .....	36
Ilustración 24. Comando uname .....	37
Ilustración 25. Comando pstree .....	37
Ilustración 26. Comando man .....	38
Ilustración 27. Comando mkdir .....	38
Ilustración 28. Comando w .....	38
Ilustración 29. Comando kill -9 .....	39
Ilustración 30. Función cat .....	39
Ilustración 31. Comando pico antes de compilarse .....	39
Ilustración 32. Comando pico después de compilarse .....	40
Ilustración 33. Comando who .....	40
Ilustración 34. Comando fg .....	40
Ilustración 35. Comando nano antes de ejecutarse .....	40
Ilustración 36. Comando nano después de ejecutarse .....	40
Ilustración 37. Comando pwd .....	40
Ilustración 38. Comando cd .....	40
Ilustración 39. Comando vi antes de ejecutarse .....	41
Ilustración 40. Comando vi después de ejecutarse .....	41
Ilustración 41. Comando wc .....	41
Ilustración 42. Comando ls .....	41
Ilustración 43. Comando apt-get .....	42
Ilustración 44. Comando sudo .....	42
Ilustración 45. Ruta relativa al escritorio .....	43
Ilustración 46. Ruta absoluta al escritorio .....	43
Ilustración 47. Carpetas .....	43
Ilustración 48. Ruta absoluta a las carpetas .....	44
Ilustración 49. Rutas relativas a las carpetas .....	44

Ilustración 50. Creación de carpetas.....	44
Ilustración 51. Ver las carpetas .....	44
Ilustración 52. Ruta absoluta a la carpeta 5 .....	44
Ilustración 53. Ruta relativa a la carpeta 10.....	45
Ilustración 54. Ruta relativa a la carpeta 9.....	45
Ilustración 55. Carpetas.....	45
Ilustración 56. Borrado con rm.....	46
Ilustración 57. Archivos de la carpeta 3 .....	46
Ilustración 58. Borrado con comodín* .....	46
Ilustración 59. Archivos con nombre similar.....	47
Ilustración 60. Borrado con rm usando comodín? .....	47
Ilustración 61. Archivos que terminan en la misma letra.....	47
Ilustración 62. Borrado con rm usando??? .....	47
Ilustración 63. Redireccionamiento > de un comando a un archivo .....	48
Ilustración 64. Mostrando el contenido del archivo.....	48
Ilustración 65. Redireccionamiento > de otro comando a un archivo .....	48
Ilustración 66. Ejecución del comando sin redireccionar .....	49
Ilustración 67. Redireccionamiento usando >> .....	49
Ilustración 68. Redireccionamiento de otro comando usando >>.....	50
Ilustración 69. Instalar en línea de comandos del paquete gimp .....	50
Ilustración 70. Comando para instalar gimp .....	51
Ilustración 71. Instalar slack.....	51
Ilustración 73. Ejemplo1 antes de ejecutar el comando- consola.....	52
Ilustración 72. Ejemplo1 antes de ejecutar el comando-carpeta .....	52
Ilustración 74. Ejemplo1 comando ejecutado-consola.....	52
Ilustración 75. Ejemplo1 comando ejecutado-carpeta .....	52
Ilustración 76. Ejemplo2 antes de ejecutar el comando- consola.....	53
Ilustración 77. Ejemplo2 antes de ejecutar el comando-carpeta .....	53
Ilustración 79. Ejemplo2 comando ejecutado-consola.....	53
Ilustración 78. Ejemplo2 comando ejecutado-carpeta .....	53
Ilustración 81. Ejemplo3 antes de ejecutar el comando-consola.....	54
Ilustración 80. Antes de ejecutar el comando-carpeta .....	54
Ilustración 82. Ejemplo3 comando ejecutado-consola.....	54
Ilustración 83. Ejemplo3 comando ejecutado-carpeta .....	54
Ilustración 84. Ejemplo4 antes de ejecutar el comando .....	55
Ilustración 85. Ejemplo4 comando ejecutado .....	55
Ilustración 87. Ejemplo5 comando ejecutado .....	56
Ilustración 86. Ejemplo5 antes de ejecutar el comando .....	56
Ilustración 88. Super usuario.....	57
Ilustración 89. Agregando un nuevo usuario .....	57
Ilustración 90. Agregando un nuevo usuario, confirmación .....	58
Ilustración 91. Creando un nuevo grupo.....	58
Ilustración 92. Creando un nuevo usuario a partir del grupo .....	59
Ilustración 93. Verificar usuarios forma gráfica .....	59
Ilustración 94. Verificar usuarios desde terminal .....	59
Ilustración 95. Ejecución de la verificación.....	60
Ilustración 96. Usuarios con sus atributos .....	60
Ilustración 97. Verificación de usuarios con ls .....	60
Ilustración 98. Cambiar de usuario.....	61
Ilustración 99. Eliminar un usuario .....	61
Ilustración 100. Verificación de eliminar un usuario .....	62

Ilustración 101. Creación de un Hola Mundo .....	62
Ilustración 102. Código Hola mundo editado en Nano .....	62
Ilustración 103. Compilación nano del código Hola Mundo .....	62
Ilustración 104. Creación Hola mundo para vi .....	63
Ilustración 105. Edición Hola Mundo en vi .....	63
Ilustración 106. Ejecución Hola Mundo en vi .....	63
Ilustración 107. Instalación de pico .....	63
Ilustración 108. Comandos de instalación pico .....	64
Ilustración 109. Actualización del apt .....	64
Ilustración 110. Soporte para pico .....	65
Ilustración 111. Pico Distribution/Download .....	65
Ilustración 112. Instalación de emacs .....	66
Ilustración 113. Instalación de emacs comando .....	66
Ilustración 114. Código del programa Hola mundo .....	66
Ilustración 115. Ejecución del programa dentro de Emacs .....	67
Ilustración 116. Compilación Programa1a.c .....	67
Ilustración 117. Compilación y pantalla de programa1a.c .....	67
Ilustración 118. Compilación y pantalla de programa2a .....	68
Ilustración 119. Pantalla de programa2a .....	68
Ilustración 120. Compilación y pantalla de programa3a .....	70

## Índice de tablas

Tabla 1. Operadores de redireccionamiento .....	9
Tabla 2. Lista de comandos .....	29

## **Glosario de términos.**

### **Máquina virtual**

Es un software que simula un sistema de computación y puede ejecutar programas como si fuese una computadora real. Este software en un principio fue definido como "un duplicado eficiente y aislado de una máquina física".

# Contenido

## Escritorios GNOME y KDE

Todos nosotros hemos usado al menos una vez una computadora, pero pocas veces nos hemos preguntado porque esta es tan sencilla de usar. Esta facilidad se debe a los entornos de escritorio que en si son un conjunto de Software (Programas y/o Aplicaciones) que ayuda a que el usuario de una computadora interactúe cómodamente con esta de una forma amigable.

Entre los populares esta KDE, un proyecto internacional de software libre, orientado específicamente a desarrollar un entorno de escritorio e infraestructura de desarrollo para los sistemas operativos GNU/Linux, Mac OSX e incluso, Windows.

KDE es un entorno de escritorio moderno, y que según su propia página [1] ofrece un extraordinario soporte para hasta 75 idiomas diferentes, lo que ofrece a los usuarios un nivel de comprensión del entorno mucho más adaptado a su idioma.

Aun así, este software también tiene competencia siendo comparado normalmente con GNOME (GNU Network Object Model Environment /Entorno de Modelo de Objeto de Red GNU), otro entorno de escritorio e infraestructura de desarrollo nacido en 1997 como alternativa del propio KDE ,para sistemas operativos GNU/Linux, Unix y derivados Unix como BSD o Solaris; compuesto enteramente de software libre.

GNOME está disponible en las principales distribuciones GNU/Linux, incluyendo Fedora, Debian, Ubuntu, Manjaro Linux, Red Hat Enterprise Linux, SUSE Linux Enterprise, CentOS, Oracle Linux, Arch Linux, Gentoo, SteamOS, entre otras. También, se encuentra disponible en Solaris, un importante sistema operativo UNIX y en Sistemas operativos Unix-like como FreeBSD. Todo acorde a la información de su sitio oficial. [2]

Esta rivalidad entre ambos softwares ha variado con el tiempo al grado de que muchas veces se ha rumorado dentro de sus comunidades de una colaboración de ambos para crear una fusión de entornos de escritorio pero ninguno se ha concretado.

## **Entornos Command Line Interface y Graphical User Interface**

Una computadora no sirve de mucho si no se puede comunicar uno con ella, por eso es que desde el inicio de estas se ha buscado diferentes formas de poder lograr esto, siendo una de las más exitosas las interfaces de línea de comando o interfaces de línea de órdenes (en inglés, command-line interfaces, CLI). Estos son métodos que permite a los usuarios dar instrucciones a algún programa informático por medio de una línea de texto simple. No debe confundirse los conceptos de CLI, shell y emulador de terminal, porque no son lo mismo ya que un CLI es un método mientras que un shell y emulador de terminal son programas informáticos. [3]

Las CLI pueden emplearse interactivamente, escribiendo instrucciones en alguna especie de entrada de texto, o pueden utilizarse de una forma mucho más automatizada (archivo batch), leyendo órdenes desde un archivo de scripts.

Esta interfaz existe casi desde los comienzos de la computación, superada en antigüedad solo por las tarjetas perforadas y mecanismos similares. Existen para diversos programas y sistemas operativos, para diverso hardware, y con distinta funcionalidad.

Por ejemplo, las CLI son parte fundamental de los shells o emuladores de terminal. Aparecen en todas las interfaces de escritorio (GNOME, KDE, Microsoft Windows) como un método para ejecutar aplicaciones rápidamente. También se utilizan en aplicaciones cliente-servidor, en gestores de bases de datos, en clientes FTP, etc. Las CLI son además un elemento fundamental de aplicaciones de ingeniería tan importantes como MATLAB y AutoCAD.

Otra forma que también ha tenido mucho éxito en la forma en la cual se comunica los usuarios con la maquina son a aquellas que apelan más a la parte gráfica. Esta orientación tiene sentido ya que al ser el humano una criatura muy visual, este puede entender mejor como proceder en diferentes situaciones si hay un apoyo grafico que le ayude, además de que una buena selección de diseños y colores puede hacer que el usuario pueda comunicarse mejor con la máquina.

Estos programas son conocidos como interfaces gráficas de usuario, conocidas también como GUI (del inglés graphical user interfaces), que cumplen con lo mencionado anteriormente, siendo un programa informático, que utiliza un conjunto de imágenes y objetos gráficos para representar la información y acciones disponibles en la interfaz. Su principal uso, consiste en proporcionar un

entorno visual sencillo para permitir la comunicación con el sistema operativo de una máquina o computador.

Habitualmente las acciones se realizan mediante manipulación directa, para facilitar la interacción del usuario con la computadora. Surge como evolución de las interfaces de línea de comandos que se usaban para operar los primeros sistemas operativos y es pieza fundamental en un entorno gráfico. Como ejemplos de interfaz gráfica de usuario, cabe citar los entornos de escritorio Windows, el X-Windows de GNU/Linux o el de Mac OS X, Aqua. [4]

En el contexto del proceso de interacción persona-computadora, la interfaz gráfica de usuario es el artefacto tecnológico de un sistema interactivo que posibilita, a través del uso y la representación del lenguaje visual, una interacción amigable con un sistema informático.

## **Terminal de Linux.**

Como todos los sistemas operativos basados en Unix , Linux disponen de un intérprete de comandos u órdenes (conocido como terminal, consola, Shell o bash ) que hace de interfaz entre el usuario y el propio sistema operativo. La terminal o consola es una herramienta poderosa capaz de realizar tareas de forma más rápida que mediante la interfaz gráfica del sistema operativo. [5]

Si bien es cierto, que para un usuario normal no tiene mucha relevancia el aprenderlo, para un usuario que quiere convertirse en un desarrollador o administrador de sistemas operativos cobra gran importancia debido a que va a tener que utilizar esta herramienta constantemente, ya sea para navegar por el sistema/directorios de forma rápida y sencilla, iniciar o detener procesos, instalar librerías, etc. Un terminal es una forma de acceder al sistema sin utilizar la interfaz gráfica, es decir, realizar todo tipo de tareas en formato texto. La forma de utilizar el sistema de este modo es mediante órdenes.

El terminal muestra en pantalla un indicador de línea de órdenes (en inglés se utiliza la palabra prompt que literalmente traduciría "prontuario" pero se puede definir como ayuda visual o palabra que ayuda) esperando que el usuario introduzca una orden. Este indicador finaliza generalmente por un carácter \$, cuando eres un usuario normal, o # cuando se es un súper usuario (administrador).

Para acceder a una terminal se puede hacer de dos formas, una es con una aplicación como el terminal de GNOME, xterm o konsole de KDE, que son emuladores de la terminal dentro de una interfaz

visual. Otra forma es salirse del entorno gráfico y acceder a un entorno completamente en modo texto, algo así como entrar en sólo símbolo de sistema en Windows 98.

Tomando como base a Ellen Seiver [6] cada vez que se ejecuta una orden, el sistema operativo le abre automáticamente tres interfaces (en los sistemas operativos tipo UNIX se utiliza el término archivo): la entrada estándar, la salida estándar y el error estándar.

- La entrada estándar (stdin) se refiere al archivo por el que una orden recibe su entrada (por defecto, es el teclado).
- La salida estándar (stdout) se refiere al archivo por el que una orden presenta sus resultados (por defecto, es la pantalla o más concretamente la ventana en la que se está ejecutando el intérprete de órdenes).
- El error estándar (stderr) se refiere al archivo por el que una orden presenta los mensajes que va generando cuando ocurre un error (por defecto, también es la pantalla).

## Tipos de usuario en Linux

Todo sistema que este pensado para tener más de un usuario (multiusuario) tiene que darles diferentes características y capacidades a estos ya que si no se hiciera estos podrían traer un enorme problema de seguridad. Dentro de Linux se puede identificar tres tipos de usuarios en Linux [7]:

- Usuario ROOT: También llamado super-usuario o administrador. Es el único usuario con privilegios sobre el control total del sistema, y por tanto, el único que tiene derecho a administrar las cuentas de usuario del sistema.
- Usuarios normales o finales: Son los usuarios habituales del sistema, que utilizarán los recursos de éste. Cada usuario sólo podrá personalizar su entorno de trabajo.
- Usuarios especiales o de sistema: Son incorporados por el propio sistema, y se encargan de administrar los demonios del mismo. Estos usuarios no pueden iniciar sesión en el sistema ni tener un Shell donde trabajar, por tanto no tienen contraseña asignada. Ejemplos son bin, daemon, adm, lp, sync, shutdown, mail, operator, squid, apache, etc.

Linux permite además la creación de grupos de usuarios, para poder establecer permisos y restricciones por grupos en lugar de por usuarios individuales.

Por defecto, cada usuario pertenecerá a un grupo que se crea de forma automática con su mismo nombre de usuario. Será el usuario root quien que pueda asignar los usuarios a los distintos grupos del sistema.

En el sistema existen una serie de grupos predefinidos por defecto. Cada usuario posee un directorio personal o directorio HOME en el que se almacena la configuración de su entorno o perfil así como los datos personales de dicho usuario. Por defecto, a ese directorio sólo tendrá acceso el mismo usuario, a no ser que él específicamente conceda permisos a otros usuarios para poder acceder.

Linux identifica a los usuarios y grupos creados en el sistema por un número, único para cada uno de ellos, y no por el nombre que le pongamos al usuario o al grupo. Lo asigna el sistema automáticamente, aunque el usuario root también lo puede establecer manualmente.

## Direccionamiento relativo y absoluto.

Cuando se trabaja con comandos es habitual tener que pasar como parámetros archivos o directorios. Para indicar un archivo o directorio se utiliza la ruta o path, que puede ser absoluta o relativa. [8]

Las rutas relativas indican el camino para encontrar un elemento, pero basándose en el directorio desde el que se ejecuta la orden, es decir, desde el directorio en donde se encuentran posicionados. En la imagen 1 se puede observar mejor esto mucho mejor.

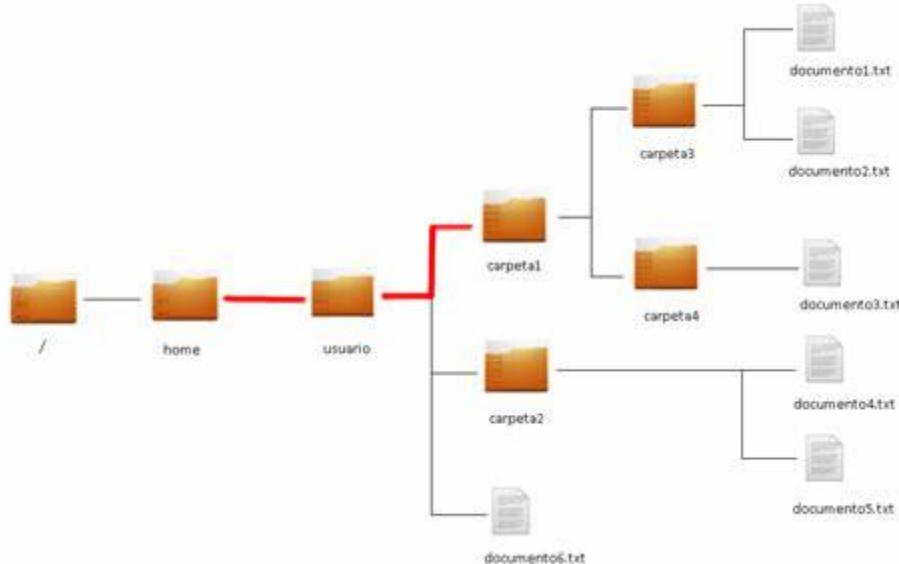


Ilustración 1. Ruta relativa

Si se encuentra en el directorio /home y se quisiera mover al directorio /home/usuario/carpeta1 se debería indicar el camino hasta donde se quiere ir partiendo desde donde se encuentra.

Por otro lado las rutas absolutas son rutas que no dependen del lugar desde donde se esté llamando una acción. El sistema de ficheros es una estructura jerárquica que en el caso de Linux tiene una raíz que se indica cuando se pone solamente el carácter barra /. La raíz contiene los directorios principales del sistema que a su vez tendrán subdirectorios en su interior, esta estructura de directorios se corresponde con el estándar de jerarquías de directorios FHS [9].

Un ejemplo de esta forma de direccionamiento se puede ejemplificar en la imagen 2. Suponiendo que se está en el directorio /home y que se quiere mover hacia el directorio /home/cristian/carpeta1 la forma de hacer esto usando una ruta absoluta sería indicando el camino desde el directorio raíz (/) hasta donde se encuentra el directorio destino

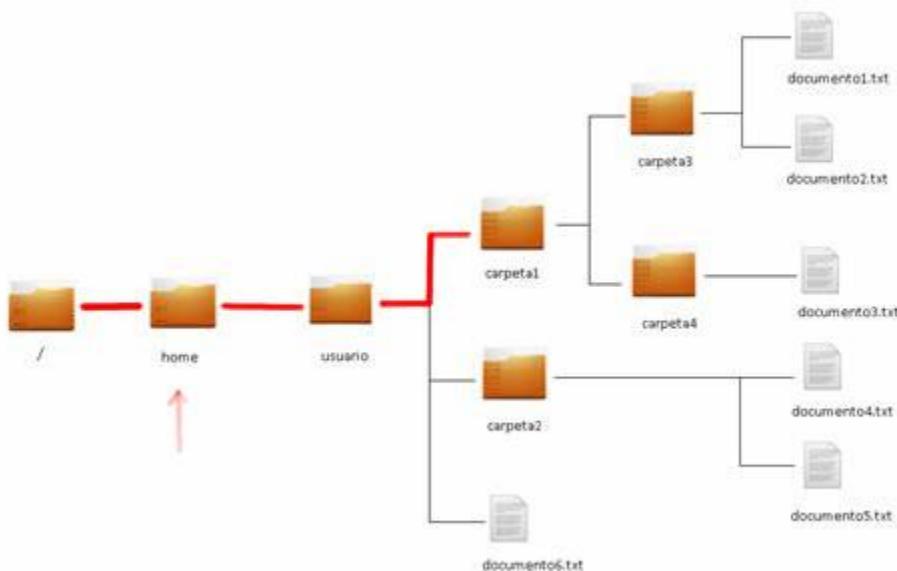


Ilustración 2. Ruta absoluta

Es muy recomendable usar estas rutas sobre las relativas ya que a pesar de que suelen ser bastante largas, tienen como ventaja que funcionan siempre, independientemente del lugar desde el que se ejecute la orden.

## Redireccionamiento.

Presentamos información sobre rediccionamiento basada en lo que menciona S.Gonzalez D. [10].

En Linux cualquier comando que se teclea, utiliza la entrada estándar [STDIN].

La salida o resultado del comando se conoce como la salida estándar [STDOUT] y es generalmente la terminal.

Sí hay errores en la ejecución estos se conocen como la salida de error estándar [STDERR] y es también por default la terminal.

Es posible y necesario además redireccionar la entrada y la salida sobre archivos en vez de utilizar la entrada y salida estándar.

Los podemos clasificar de la siguiente manera:

### FILE DESCRIPTORS – FD (Descriptores de archivos)

- STDIN Dispositivo de entrada estándar, por defecto el teclado.
- STDOUT Dispositivo de salida estándar, por defecto la terminal o pantalla.
- STDERR Dispositivo de salida de mensajes de error estándar, por defecto la terminal o pantalla.

### REDIRECTION OPERATORS (Operadores de rediccionamiento)

A continuación se muestra una tabla donde se detallan cada uno de estos operadores.

Tabla de operadores de rediccionamiento.

Símbolo	Descripción
>	Redirecciona <b>stdout</b> hacia un archivo. Lo crea si no existe, si existe lo sobrescribe.
	ls -l > lista.txt (La salida del comando se envía a un archivo en vez de la terminal.)
>>	Redirecciona <b>stdout</b> hacia un archivo. Lo crea si no existe, si existe concatena la salida al final de este.
	ps -ef >> procesos.txt (Concatena al archivo procesos.txt la salida del comando.)
<	Redirecciona <b>stdin</b> desde un archivo. El contenido de un archivo es la entrada o input del comando.
	mail user < texto.txt (El cuerpo del correo a enviar proviene desde un archivo, en vez del teclado.)
2> 2>>	Redirecciona <b>stderr</b> hacia un archivo. Crea (>) o concatena (>>) la salida de errores a un archivo. (ver ejemplos)
I>&2	Redirecciona <b>stdout</b> hacia donde <b>stderr</b> apunte. (ver ejemplos)
2>&1	Redirecciona <b>stderr</b> hacia donde <b>stdout</b> apunte. (ver ejemplos)

OTROS REDIRECCIONAMIENTOS QUE NO UTILIZAN FDs	
<<	Conocido como <b>HERE-DOCUMENT</b> o <b>HereDoc</b> (ver ejemplos)
<<<	Conocido como <b>HERE-STRING</b> (ver ejemplos)
/	El símbolo / (pipe) es un tipo de redireccionamiento ya que la salida ( <b>stdout</b> ) de un comando es la entrada ( <b>stdin</b> ) de otro. ls /etc   grep services (La salida del comando a la izquierda de / se convierte en la entrada del comando a la derecha.)
tee	El comando tee redirecciona la salida ( <b>stdout</b> ) a ambos, un archivo y a la terminal. ps -ef   tee procesos.txt (La salida de ps se muestra en la terminal y al mismo tiempo se redirecciona al archivo <i>procesos.txt</i> . Con la opción -a (tee -a) concatena al archivo.)

Tabla 1. Operadores de redireccionamiento

## Clasificación de los comandos en Linux.

A continuación presentamos una pequeña lista de comandos de Linux. [11]

### Información del sistema

- arch: mostrar la arquitectura de la máquina (primero).
- uname -m: mostrar la arquitectura de la máquina (segundo).
- uname -r: mostrar la versión del kernel usado.
- dmidecode -q: mostrar los componentes (hardware) del sistema.
- hdparm -i /dev/hda: mostrar las características de un disco duro.
- hdparm -T /dev/sda: realizar prueba de lectura en un disco duro.
- cat /proc/cpuinfo: mostrar información de la CPU.
- cat /proc/interrupts: mostrar las interrupciones.
- cat /proc/meminfo: verificar el uso de memoria.
- cat /proc/swaps: mostrar ficheros swap.
- cat /proc/version: mostrar la versión del kernel.
- cat /proc/net/dev: mostrar adaptadores de red y estadísticas.
- cat /proc/mounts: mostrar el sistema de ficheros montado.
- lspci -tv: mostrar los dispositivos PCI.
- lsusb -tv: mostrar los dispositivos USB.
- date: mostrar la fecha del sistema.
- date 041217002011.00: colocar (declarar, ajustar) fecha y hora.
- clock -w: guardar los cambios de fecha en la BIOS.

## Apagar (Reiniciar Sistema o Cerrar Sesión)

- shutdown -h now: apagar el sistema (1).
- init 0: apagar el sistema (2).
- telinit 0: apagar el sistema (3).
- halt: apagar el sistema (4).
- shutdown -h hours:minutes &: apagado planificado del sistema.
- shutdown -c: cancelar un apagado planificado del sistema.
- shutdown -r now: reiniciar (1).
- reboot: reiniciar (2).
- logout: cerrar sesión.

## Archivos y Directorios

- cd /home: entrar en el directorio “home”.
- cd ..: retroceder un nivel.
- cd ../../: retroceder 2 niveles.
- cd: ir al directorio raíz.
- cd ~user1: ir al directorio user1.
- cd -: ir (regresar) al directorio anterior.
- pwd: mostrar el camino del directorio de trabajo.
- ls: ver los ficheros de un directorio.
- ls -F: ver los ficheros de un directorio.
- ls -l: mostrar los detalles de ficheros y carpetas de un directorio.
- ls -a: mostrar los ficheros ocultos.
- ls \*[0-9]\*: mostrar los ficheros y carpetas que contienen números.
- tree: mostrar los ficheros y carpetas en forma de árbol comenzando por la raíz.(1)
- lmtree: mostrar los ficheros y carpetas en forma de árbol comenzando por la raíz.(2)
- mkdir dir1: crear una carpeta o directorio con nombre ‘dir1’.
- mkdir dir1 dir2: crear dos carpetas o directorios simultáneamente (Crear dos directorios a la vez).

- mkdir -p /tmp/dir1/dir2: crear un árbol de directorios.
- rm -f file1: borrar el fichero llamado ‘file1’.
- rmdir dir1: borrar la carpeta llamada ‘dir1’.
- rm -rf dir1: eliminar una carpeta llamada ‘dir1’ con su contenido de forma recursiva. (Si lo borro recursivo estoy diciendo que es con su contenido).
- rm -rf dir1 dir2: borrar dos carpetas (directorios) con su contenido de forma recursiva.
- mv dir1 new\_dir: renombrar o mover un fichero o carpeta (directorio).
- cp file1: copiar un fichero.
- cp file1 file2: copiar dos ficheros al unísono.
- cp dir /\* .: copiar todos los ficheros de un directorio dentro del directorio de trabajo actual.
- cp -a /tmp/dir1 .: copiar un directorio dentro del directorio actual de trabajo.
- cp -a dir1: copiar un directorio.
- cp -a dir1 dir2: copiar dos directorio al unísono.
- ln -s file1 lnk1: crear un enlace simbólico al fichero o directorio.
- ln file1 lnk1: crear un enlace físico al fichero o directorio.
- touch -t 0712250000 file1: modificar el tiempo real (tiempo de creación) de un fichero o directorio.
- file file1: salida (volcado en pantalla) del tipo mime de un fichero texto.
- iconv -l: listas de cifrados conocidos.
- iconv -f fromEncoding -t toEncoding inputFile > outputFile: crea una nueva forma del fichero de entrada asumiendo que está codificado en fromEncoding y convirtiéndolo a ToEncoding.
- find . -maxdepth 1 -name \*.jpg -print -exec convert "{}" -resize 80×60 "thumbs/{}" \;: agrupar ficheros redimensionados en el directorio actual y enviarlos a directorios en vistas de miniaturas (requiere convertir desde ImagemagicK).

## Encontrar archivos

- find / -name file1: buscar fichero y directorio a partir de la raíz del sistema.
- find / -user user1: buscar ficheros y directorios pertenecientes al usuario ‘user1’.
- find /home/user1 -name \*.bin: buscar ficheros con extensión ‘. bin’ dentro del directorio ‘/ home/user1’.
- find /usr/bin -type f -atime +100: buscar ficheros binarios no usados en los últimos 100 días.

- find /usr/bin -type f -mtime -10: buscar ficheros creados o cambiados dentro de los últimos 10 días.
- find / -name \\*.rpm -exec chmod 755 '{}' \;: buscar ficheros con extensión ‘.rpm’ y modificar permisos.
- find / -xdev -name \\*.rpm: Buscar ficheros con extensión ‘.rpm’ ignorando los dispositivos removibles como cdrom, pen-drive, etc....
- locate \\*.ps: encuentra ficheros con extensión ‘.ps’ ejecutados primeramente con el command ‘updatedb’.
- whereis halt: mostrar la ubicación de un fichero binario, de ayuda o fuente. En este caso pregunta dónde está el comando ‘halt’.
- which halt: mostrar la senda completa (el camino completo) a un binario / ejecutable.

## Montando un sistema de ficheros.

- mount /dev/hda2 /mnt/hda2: montar un disco llamado hda2. Verifique primero la existencia del directorio ‘/ mnt/hda2’; si no está, debe crearlo.
- umount /dev/hda2: desmontar un disco llamado hda2. Salir primero desde el punto ‘/ mnt/hda2’.
- fuser -km /mnt/hda2: forzar el desmontaje cuando el dispositivo está ocupado.
- umount -n /mnt/hda2: correr el desmontaje sin leer el fichero /etc/mtab. Útil cuando el fichero es de solo lectura o el disco duro está lleno.
- mount /dev/fd0 /mnt/floppy: montar un disco flexible (floppy).
- mount /dev/cdrom /mnt/cdrom: montar un cdrom / dvdrom.
- mount /dev/hdc /mnt/cdrecorder: montar un cd regrabable o un dvdrom.
- mount /dev/hdb /mnt/cdrecorder: montar un cd regrabable / dvdrom (un dvd).
- mount -o loop file.iso /mnt/cdrom: montar un fichero o una imagen iso.
- mount -t vfat /dev/hda5 /mnt/hda5: montar un sistema de ficheros FAT32.
- mount /dev/sda1 /mnt/usbdisk: montar un usb pen-drive o una memoria (sin especificar el tipo de sistema de ficheros).

## Espacio de Disco

- df -h: mostrar una lista de las particiones montadas.

- ls -lSr |more: mostrar el tamaño de los ficheros y directorios ordenados por tamaño.
- du -sh dir1: Estimar el espacio usado por el directorio ‘dir1’.
- du -sk \* | sort -rn: mostrar el tamaño de los ficheros y directorios ordenados por tamaño.
- rpm -q -a --queryformat ‘%10{SIZE}t%{NAME}n’ | sort -k1,1n: mostrar el espacio usado por los paquetes rpm instalados organizados por tamaño (Fedora, Redhat y otros).
- dpkg-query -W -f='\${Installed-Size;10}t\${Package}n' | sort -k1,1n: mostrar el espacio usado por los paquetes instalados, organizados por tamaño (Ubuntu, Debian y otros).

## Usuarios y Grupos

- groupadd nombre\_del\_grupo: crear un nuevo grupo.
- groupdel nombre\_del\_grupo: borrar un grupo.
- groupmod -n nuevo\_nombre\_del\_grupo viejo\_nombre\_del\_grupo: renombrar un grupo.
- useradd -c “Name Surname” -g admin -d /home/user1 -s /bin/bash user1: Crear un nuevo usuario perteneciente al grupo “admin”.
- useradd user1: crear un nuevo usuario.
- userdel -r user1: borrar un usuario (‘-r’ elimina el directorio Home).
- usermod -c “User FTP” -g system -d /ftp/user1 -s /bin/nologin user1: cambiar los atributos del usuario.
- passwd: cambiar contraseña.
- passwd user1: cambiar la contraseña de un usuario (solamente por root).
- chage -E 2011-12-31 user1: colocar un plazo para la contraseña del usuario. En este caso dice que la clave expira el 31 de diciembre de 2011.
- pwck: chequear la sintaxis correcta el formato de fichero de ‘/etc/passwd’ y la existencia de usuarios.
- grpck: chequear la sintaxis correcta y el formato del fichero ‘/etc/group’ y la existencia de grupos.
- newgrp group\_name: registra a un nuevo grupo para cambiar el grupo predeterminado de los ficheros creados recientemente.

Permisos en Ficheros (Usa “+” para colocar permisos y “-” para eliminar)

- ls -lh: Mostrar permisos.

- ls /tmp | pr -T5 -W\$COLUMNS: dividir la terminal en 5 columnas.
- chmod ugo+rwx directory1: colocar permisos de lectura (r), escritura (w) y ejecución(x) al propietario (u), al grupo (g) y a otros (o) sobre el directorio ‘directory1’.
- chmod go-rwx directory1: quitar permiso de lectura (r), escritura (w) y (x) ejecución al grupo (g) y otros (o) sobre el directorio ‘directory1’.
- chown user1 file1: cambiar el dueño de un fichero.
- chown -R user1 directory1: cambiar el propietario de un directorio y de todos los ficheros y directorios contenidos dentro.
- chgrp group1 file1: cambiar grupo de ficheros.
- chown user1:group1 file1: cambiar usuario y el grupo propietario de un fichero.
- find / -perm -u+s: visualizar todos los ficheros del sistema con SUID configurado.
- chmod u+s /bin/file1: colocar el bit SUID en un fichero binario. El usuario que corriendo ese fichero adquiere los mismos privilegios como dueño.
- chmod u-s /bin/file1: deshabilitar el bit SUID en un fichero binario.
- chmod g+s /home/public: colocar un bit SGID en un directorio –similar al SUID pero por directorio.
- chmod g-s /home/public: desabilitar un bit SGID en un directorio.
- chmod o+t /home/public: colocar un bit STIKY en un directorio. Permite el borrado de ficheros solamente a los dueños legítimos.
- chmod o-t /home/public: desabilitar un bit STIKY en un directorio.

Atributos especiales en ficheros (Usa "+" para colocar permisos y "-" para eliminar)

- chattr +a file1: permite escribir abriendo un fichero solamente modo append.
- chattr +c file1: permite que un fichero sea comprimido / descomprimido automaticamente.
- chattr +d file1: asegura que el programa ignore borrar los ficheros durante la copia de seguridad.
- chattr +i file1: convierte el fichero en invariable, por lo que no puede ser eliminado, alterado, renombrado, ni enlazado.
- chattr +s file1: permite que un fichero sea borrado de forma segura.
- chattr +S file1: asegura que un fichero sea modificado, los cambios son escritos en modo synchronous como con sync.
- chattr +u file1: te permite recuperar el contenido de un fichero aún si este está cancelado.

- lsattr: mostrar atributos especiales.

## Archivos y Ficheros comprimidos.

- bunzip2 file1.bz2: descomprime un fichero llamado ‘file1.bz2’.
- bzip2 file1: comprime un fichero llamado ‘file1’.
- gunzip file1.gz: descomprime un fichero llamado ‘file1.gz’.
- gzip file1: comprime un fichero llamado ‘file1’.
- gzip -9 file1: comprime con compresión máxima.
- rar a file1.rar test\_file: crear un fichero rar llamado ‘file1.rar’.
- rar a file1.rar file1 file2 dir1: comprimir ‘file1’, ‘file2’ y ‘dir1’ simultáneamente.
- rar x file1.rar: descomprimir archivo rar.
- unrar x file1.rar: descomprimir archivo rar.
- tar -cvf archive.tar file1: crear un tarball descomprimido.
- tar -cvf archive.tar file1 file2 dir1: crear un archivo contenido ‘file1’, ‘file2’ y ‘dir1’.
- tar -tf archive.tar: mostrar los contenidos de un archivo.
- tar -xvf archive.tar: extraer un tarball.
- tar -xvf archive.tar -C /tmp: extraer un tarball en / tmp.
- tar -cvfj archive.tar.bz2 dir1: crear un tarball comprimido dentro de bzip2.
- tar -xvfj archive.tar.bz2: descomprimir un archivo tar comprimido en bzip2
- tar -cvfz archive.tar.gz dir1: crear un tarball comprimido en gzip.
- tar -xvfz archive.tar.gz: descomprimir un archive tar comprimido en gzip.
- zip file1.zip file1: crear un archivo comprimido en zip.
- zip -r file1.zip file1 file2 dir1: comprimir, en zip, varios archivos y directorios de forma simultánea.
- unzip file1.zip: descomprimir un archivo zip.

## Paquetes RPM (Red Hat, Fedora y similares).

- rpm -ivh package.rpm: instalar un paquete rpm.
- rpm -ivh --nodeps package.rpm: instalar un paquete rpm ignorando las peticiones de dependencias.
- rpm -U package.rpm: actualizar un paquete rpm sin cambiar la configuración de los ficheros.

- rpm -F package.rpm: actualizar un paquete rpm solamente si este está instalado.
- rpm -e package\_name.rpm: eliminar un paquete rpm.
- rpm -qa: mostrar todos los paquetes rpm instalados en el sistema.
- rpm -qa | grep httpd: mostrar todos los paquetes rpm con el nombre “httpd”.
- rpm -qi package\_name: obtener información en un paquete específico instalado.
- rpm -qg “System Environment/Daemons”: mostar los paquetes rpm de un grupo software.
- rpm -ql package\_name: mostrar lista de ficheros dados por un paquete rpm instalado.
- rpm -qc package\_name: mostrar lista de configuración de ficheros dados por un paquete rpm instalado.
- rpm -q package\_name –whatrequires: mostrar lista de dependencias solicitada para un paquete rpm.
- rpm -q package\_name –whatprovides: mostar la capacidad dada por un paquete rpm.
- rpm -q package\_name –scripts: mostrar los scripts comenzados durante la instalación /eliminación.
- rpm -q package\_name –changelog: mostar el historial de revisions de un paquete rpm.
- rpm -qf /etc/httpd/conf/httpd.conf: verificar cuál paquete rpm pertenece a un fichero dado.
- rpm -qp package.rpm -l: mostrar lista de ficheros dados por un paquete rpm que aún no ha sido instalado.
- rpm –import /media/cdrom/RPM-GPG-KEY: importar la firma digital de la llave pública.
- rpm –checksig package.rpm: verificar la integridad de un paquete rpm.
- rpm -qa gpg-pubkey: verificar la integridad de todos los paquetes rpm instalados.
- rpm -V package\_name: chequear el tamaño del fichero, licencias, tipos, dueño, grupo, chequeo de resumen de MD5 y última modificación.
- rpm -Va: chequear todos los paquetes rpm instalados en el sistema. Usar con cuidado.
- rpm -Vp package.rpm: verificar un paquete rpm no instalado todavía.
- rpm2cpio package.rpm | cpio –extract –make-directories \*bin\*: extraer fichero ejecutable desde un paquete rpm.
- rpm -ivh /usr/src/redhat/RPMS/`arch`/package.rpm: instalar un paquete construido desde una fuente rpm.
- rpmbuild –rebuild package\_name.src.rpm: construir un paquete rpm desde una fuente rpm.

## Actualizador de paquetes YUM (Red Hat, Fedora y similares)

- `yum install package_name`: descargar e instalar un paquete rpm.
- `yum localinstall package_name.rpm`: este instalará un RPM y tratará de resolver todas las dependencies para ti, usando tus repositorios.
- `yum update package_name.rpm`: actualizar todos los paquetes rpm instalados en el sistema.
- `yum update package_name`: modernizar / actualizar un paquete rpm.
- `yum remove package_name`: eliminar un paquete rpm.
- `yum list`: listar todos los paquetes instalados en el sistema.
- `yum search package_name`: Encontrar un paquete en repositorio rpm.
- `yum clean packages`: limpiar un caché rpm borrando los paquetes descargados.
- `yum clean headers`: eliminar todos los ficheros de encabezamiento que el sistema usa para resolver la dependencia.
- `yum clean all`: eliminar desde los paquetes caché y ficheros de encabezado.

### Paquetes Deb (Debian, Ubuntu y derivados).

- `dpkg -i package.deb`: instalar / actualizar un paquete deb.
- `dpkg -r package_name`: eliminar un paquete deb del sistema.
- `dpkg -l`: mostrar todos los paquetes deb instalados en el sistema.
- `dpkg -l | grep httpd`: mostrar todos los paquetes deb con el nombre “httpd”
- `dpkg -s package_name`: obtener información en un paquete específico instalado en el sistema.
- `dpkg -L package_name`: mostrar lista de ficheros dados por un paquete instalado en el sistema.
- `dpkg --contents package.deb`: mostrar lista de ficheros dados por un paquete no instalado todavía.
- `dpkg -S /bin/ping`: verificar cuál paquete pertenece a un fichero dado.

### Actualizador de paquetes APT (Debian, Ubuntu y derivados).

- `apt-get install package_name`: instalar / actualizar un paquete deb.
- `apt-cdrom install package_name`: instalar / actualizar un paquete deb desde un cdrom.
- `apt-get update`: actualizar la lista de paquetes.
- `apt-get upgrade`: actualizar todos los paquetes instalados.
- `apt-get remove package_name`: eliminar un paquete deb del sistema.
- `apt-get check`: verificar la correcta resolución de las dependencias.

- apt-get clean: limpiar cache desde los paquetes descargados.
- apt-cache search searched-package: retorna lista de paquetes que corresponde a la serie de paquetes buscados.

Ver el contenido de un fichero.

- cat file1: ver los contenidos de un fichero comenzando desde la primera hilera.
- tac file1: ver los contenidos de un fichero comenzando desde la última línea.
- more file1: ver el contenido a lo largo de un fichero.
- less file1: parecido al comando ‘more’ pero permite salvar el movimiento en el fichero así como el movimiento hacia atrás.
- head -2 file1: ver las dos primeras líneas de un fichero.
- tail -2 file1: ver las dos últimas líneas de un fichero.
- tail -f /var/log/messages: ver en tiempo real qué ha sido añadido al fichero.

Manipulación de texto.

- cat file1 file2 .. | command <> file1\_in.txt\_or\_file1\_out.txt: sintaxis general para la manipulación de texto utilizando PIPE, STDIN y STDOUT.
- cat file1 | command( sed, grep, awk, grep, etc...) > result.txt: sintaxis general para manipular un texto de un fichero y escribir el resultado en un fichero nuevo.
- cat file1 | command( sed, grep, awk, grep, etc...) » result.txt: sintaxis general para manipular un texto de un fichero y añadir resultado en un fichero existente.
- grep Aug /var/log/messages: buscar palabras “Aug” en el fichero ‘/var/log/messages’.
- grep ^Aug /var/log/messages: buscar palabras que comienzan con “Aug” en fichero ‘/var/log/messages’
- grep [0-9] /var/log/messages: seleccionar todas las líneas del fichero ‘/var/log/messages’ que contienen números.
- grep Aug -R /var/log/\*: buscar la cadena “Aug” en el directorio ‘/var/log’ y debajo.
- sed ‘s/stringa1/stringa2/g’ example.txt: reubicar “string1” con “string2” en ejemplo.txt
- sed ‘/^\$/d’ example.txt: eliminar todas las líneas en blanco desde el ejemplo.txt
- sed ‘/ \*#/d; /^\$/d’ example.txt: eliminar comentarios y líneas en blanco de ejemplo.txt
- echo ‘esempio’ | tr ‘[:lower:]’ ‘[:upper:]’: convertir minúsculas en mayúsculas.

- sed -e ‘1d’ result.txt: elimina la primera línea del fichero ejemplo.txt
- sed -n ‘/string1/p’: visualizar solamente las líneas que contienen la palabra “string1”.

Establecer carácter y conversión de ficheros.

- dos2unix filedos.txt fileunix.txt: convertir un formato de fichero texto desde MSDOS a UNIX.
- unix2dos fileunix.txt filedos.txt: convertir un formato de fichero de texto desde UNIX a MSDOS.
- recode ..HTML < page.txt > page.html: convertir un fichero de texto en html.
- recode -l | more: mostrar todas las conversiones de formato disponibles.

Análisis del sistema de ficheros.

- badblocks -v /dev/hda1: Chequear los bloques defectuosos en el disco hda1.
- fsck /dev/hda1: reparar / chequear la integridad del fichero del sistema Linux en el disco hda1.
- fsck.ext2 /dev/hda1: reparar / chequear la integridad del fichero del sistema ext 2 en el disco hda1.
- e2fsck /dev/hda1: reparar / chequear la integridad del fichero del sistema ext 2 en el disco hda1.
- e2fsck -j /dev/hda1: reparar / chequear la integridad del fichero del sistema ext 3 en el disco hda1.
- fsck.ext3 /dev/hda1: reparar / chequear la integridad del fichero del sistema ext 3 en el disco hda1.
- fsck.vfat /dev/hda1: reparar / chequear la integridad del fichero sistema fat en el disco hda1.
- fsck.msdos /dev/hda1: reparar / chequear la integridad de un fichero del sistema dos en el disco hda1.
- dosfsck /dev/hda1: reparar / chequear la integridad de un fichero del sistema dos en el disco hda1.

## Formatear un sistema de ficheros

- mkfs /dev/hda1: crear un fichero de sistema tipo Linux en la partición hda1.
- mke2fs /dev/hda1: crear un fichero de sistema tipo Linux ext 2 en hda1.
- mke2fs -j /dev/hda1: crear un fichero de sistema tipo Linux ext3 (periódico) en la partición hda1.
- mkfs -t vfat 32 -F /dev/hda1: crear un fichero de sistema FAT32 en hda1.
- fdformat -n /dev/fd0: formatear un disco flooply.
- mkswap /dev/hda3: crear un fichero de sistema swap.

## Trabajo con la SWAP.

- mkswap /dev/hda3: crear fichero de sistema swap.
- swapon /dev/hda3: activando una nueva partición swap.
- swapon /dev/hda2 /dev/hdb3: activar dos particiones swap.

## Salvas (Backup).

- dump -0aj -f /tmp/home0.bak /home: hacer una salva completa del directorio '/home'.
- dump -1aj -f /tmp/home0.bak /home: hacer una salva incremental del directorio '/home'.
- restore -if /tmp/home0.bak: restaurando una salva interactivamente.
- rsync -rogpav --delete /home /tmp: sincronización entre directorios.
- rsync -rogpav -e ssh --delete /home ip\_address:/tmp: rsync a través del túnel SSH.
- rsync -az -e ssh --delete ip\_addr:/home/public /home/local: sincronizar un directorio local con un directorio remoto a través de ssh y de compresión.
- rsync -az -e ssh --delete /home/local ip\_addr:/home/public: sincronizar un directorio remoto con un directorio local a través de ssh y de compresión.
- dd bs=1M if=/dev/hda | gzip | ssh user@ip\_addr 'dd of=hda.gz': hacer una salva de un disco duro en un host remoto a través de ssh.
- dd if=/dev/sda of=/tmp/file1: salvar el contenido de un disco duro a un fichero. (En este caso el disco duro es "sda" y el fichero "file1").
- tar -Puf backup.tar /home/user: hacer una salva incremental del directorio '/home/user'.

- ( cd /tmp/local/ && tar c . ) | ssh -C user@ip\_addr ‘cd /home/share/ && tar x -p’: copiar el contenido de un directorio en un directorio remoto a través de ssh.
- ( tar c /home ) | ssh -C user@ip\_addr ‘cd /home/backup-home && tar x -p’: copiar un directorio local en un directorio remoto a través de ssh.
- tar cf – . | (cd /tmp/backup ; tar xf – ): copia local conservando las licencias y enlaces desde un directorio a otro.
- find /home/user1 -name ‘\*.txt’ | xargs cp -av –target-directory=/home/backup/ –parents: encontrar y copiar todos los ficheros con extensión ‘.txt’ de un directorio a otro.
- find /var/log -name ‘\*.log’ | tar cv –files-from=- | bzip2 > log.tar.bz2: encontrar todos los ficheros con extensión ‘.log’ y hacer un archivo bzip.
- dd if=/dev/hda of=/dev/fd0 bs=512 count=1: hacer una copia del MBR (Master Boot Record) a un disco floppy.
- dd if=/dev/fd0 of=/dev/hda bs=512 count=1: restaurar la copia del MBR (Master Boot Record) salvada en un floppy.

## CD-ROM.

- cdrecord -v gracetime=2 dev=/dev/cdrom -eject blank=fast -force: limpiar o borrar un cd regrabable.
- mkisofs /dev/cdrom > cd.iso: crear una imagen iso de cdrom en disco.
- mkisofs /dev/cdrom | gzip > cd\_iso.gz: crear una imagen comprimida iso de cdrom en disco.
- mkisofs -J -allow-leading-dots -R -V “Label CD” -iso-level 4 -o ./cd.iso data\_cd: crear una imagen iso de un directorio.
- cdrecord -v dev=/dev/cdrom cd.iso: quemar una imagen iso.
- gzip -dc cd\_iso.gz | cdrecord dev=/dev/cdrom -: quemar una imagen iso comprimida.
- mount -o loop cd.iso /mnt/iso: montar una imagen iso.
- cd-paranoia -B: llevar canciones de un cd a ficheros wav.
- cd-paranoia - ”-3”: llevar las 3 primeras canciones de un cd a ficheros wav.
- cdrecord –scanbus: escanear bus para identificar el canal scsi.
- dd if=/dev/hdc | md5sum: hacer funcionar un md5sum en un dispositivo, como un CD.

## Trabajo con la RED ( LAN y Wi-Fi)

- ifconfig eth0: mostrar la configuración de una tarjeta de red Ethernet.
- ifup eth0: activar una interface ‘eth0’.
- ifdown eth0: deshabilitar una interface ‘eth0’.
- ifconfig eth0 192.168.1.1 netmask 255.255.255.0: configurar una dirección IP.
- ifconfig eth0 promisc: configurar ‘eth0’ en modo común para obtener los paquetes (sniffing).
- dhclient eth0: activar la interface ‘eth0’ en modo dhcp.
- route -n: mostrar mesa de recorrido.
- route add -net 0/0 gw IP\_Gateway: configurar entrada predeterminada.
- route add -net 192.168.0.0 netmask 255.255.0.0 gw 192.168.1.1: configurar ruta estática para buscar la red ‘192.168.0.0/16’.
- route del 0/0 gw IP\_gateway: eliminar la ruta estática.
- echo “1” > /proc/sys/net/ipv4/ip\_forward: activar el recorrido ip.
- hostname: mostrar el nombre del host del sistema.
- host www.example.com: buscar el nombre del host para resolver el nombre a una dirección ip(1).
- nslookup www.example.com: buscar el nombre del host para resolver el nombre a una dirección ip y viceversa(2).
- ip link show: mostrar el estado de enlace de todas las interfaces.
- mii-tool eth0: mostrar el estado de enlace de ‘eth0’.
- ethtool eth0: mostrar las estadísticas de tarjeta de red ‘eth0’.
- netstat -tup: mostrar todas las conexiones de red activas y sus PID.
- netstat -tupl: mostrar todos los servicios de escucha de red en el sistema y sus PID.
- tcpdump tcp port 80: mostrar todo el tráfico HTTP.
- iwlist scan: mostrar las redes inalámbricas.
- iwconfig eth1: mostrar la configuración de una tarjeta de red inalámbrica.
- whois www.example.com: buscar en base de datos Whois.

## Tablas IP (CORTAFUEGOS)

- iptables -t filter -L: mostrar todas las cadenas de la tabla de filtro.
- iptables -t nat -L: mostrar todas las cadenas de la tabla nat.
- iptables -t filter -F: limpiar todas las reglas de la tabla de filtro.
- iptables -t nat -F: limpiar todas las reglas de la tabla nat.

- iptables -t filter -X: borrar cualquier cadena creada por el usuario.
- iptables -t filter -A INPUT -p tcp –dport telnet -j ACCEPT: permitir las conexiones telnet para entrar.
- iptables -t filter -A OUTPUT -p tcp –dport http -j DROP: bloquear las conexiones HTTP para salir.
- iptables -t filter -A FORWARD -p tcp –dport pop3 -j ACCEPT: permitir las conexiones POP a una cadena delantera.
- iptables -t filter -A INPUT -j LOG –log-prefix “DROP INPUT”: registrando una cadena de entrada.
- iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE: configurar un PAT (Puerto de traducción de dirección) en eth0, ocultando los paquetes de salida forzada.
- iptables -t nat -A PREROUTING -d 192.168.0.1 -p tcp -m tcp –dport 22 -j DNAT –to-destination 10.0.0.2:22: redireccionar los paquetes dirigidos de un host a otro.

## Monitoreando y depurando.

- top: mostrar las tareas de Linux usando la mayoría CPU.
- ps -eafw: muestra las tareas Linux.
- ps -e -o pid,args –forest: muestra las tareas Linux en un modo jerárquico.
- pstree: mostrar un árbol sistema de procesos.
- kill -9 ID\_Processo: forzar el cierre de un proceso y terminarlo.
- kill -1 ID\_Processo: forzar un proceso para recargar la configuración.
- lsof -p \$\$: mostrar una lista de ficheros abiertos por procesos.
- lsof /home/user1: muestra una lista de ficheros abiertos en un camino dado del sistema.
- strace -c ls >/dev/null: mostrar las llamadas del sistema hechas y recibidas por un proceso.
- strace -f -e open ls >/dev/null: mostrar las llamadas a la biblioteca.
- watch -n1 ‘cat /proc/interrupts’: mostrar interrupciones en tiempo real.
- last reboot: mostrar historial de reinicio.
- lsmod: mostrar el kernel cargado.
- free -m: muestra el estado de la RAM en megabytes.
- smartctl -A /dev/hda: monitorear la fiabilidad de un disco duro a través de SMART.
- smartctl -i /dev/hda: chequear si SMART está activado en un disco duro.
- tail /var/log/dmesg: mostrar eventos inherentes al proceso de carga del kernel.

- tail /var/log/messages: mostrar los eventos del sistema.

## Comandos adicionales.

- apropos ...keyword: mostrar una lista de comandos que pertenecen a las palabras claves de un programa; son útiles cuando tú sabes qué hace tu programa, pero se desconoce el nombre del comando.
- man ping: mostrar las páginas del manual on-line; por ejemplo, en un comando ping, usar la opción ‘-k’ para encontrar cualquier comando relacionado.
- whatis ...keyword: muestra la descripción de lo que hace el programa.
- mkbootdisk –device /dev/fd0 `uname -r`: crear un floppy booteable.
- gpg -c file1: codificar un fichero con guardia de seguridad GNU.
- gpg file1.gpg: decodificar un fichero con Guardia de seguridad GNU.
- wget -r www.example.com: descargar un sitio web completo.
- wget -c www.example.com/file.iso: descargar un fichero con la posibilidad de parar la descargar y reanudar más tarde.
- echo ‘wget -c www.example.com/files.iso‘ | at 09:00: Comenzar una descarga a cualquier hora. En este caso empezaría a las 9 horas.
- ldd /usr/bin/ssh: mostrar las bibliotecas compartidas requeridas por el programa ssh.
- alias hh='history': colocar un alias para un comando –hh= Historial.
- chsh: cambiar el comando Shell.
- chsh –list-shells: es un comando adecuado para saber si tienes que hacer remoto en otra terminal.
- who -a: mostrar quien está registrado, e imprimir hora del último sistema de importación, procesos muertos, procesos de registro de sistema, procesos activos producidos por init, funcionamiento actual y últimos cambios del reloj del sistema.

## Variables de entorno.

[12] Las variables de entorno son valores dinámicos que afectan a los programas o procesos que se ejecutan en un servidor. Existen en todos los sistemas operativos y su tipo puede variar. Las variables de entorno se pueden crear, editar, guardar y eliminar.

En Linux, las variables de entorno son marcadores de posición para la información almacenada dentro del sistema que pasa datos a los programas iniciados en shells (intérpretes de comando) o sub-shells. [13] Las variables de entorno se utilizan para almacenar algunos valores que pueden ser utilizados por los scripts desde el shell. Este sistema operativo nos permite crear, crear nuestras propias variables.

El shell de bash tiene dos tipos de variables de entorno:

### Variables globales

El sistema Linux establece algunas variables de entorno globales cuando inicias sesión en tu sistema y siempre son en letras mayúsculas para diferenciarlas de las variables de entorno definidas por el usuario podemos verlas, escribiendo el comando printenv. Podemos imprimir solo una de ellas, escribiendo el comando echo seguido de \$NombreDeVariable.

### Variables locales.

Linux también define algunas variables de entorno locales estándar de forma predeterminada.

Para ver las variables globales y locales del shell que está ejecutando y las que están disponible para ese shell, escribe el comando set.

### Declarar variables locales

Para declarar tus propias variables de entorno directamente desde el shell, escribe el nombreDeVariable que deseas, seguido de un signo igual y el valor SIN espacios:

miVariable=Hola

Si quieres que tu variable sea una cadena solo debes poner su valor entre comillas  
miVariable='Hola a todos'

### Declarar variables globales (variables de exportación).

Para declarar una variable de entorno global, debes declarar una variable de entorno local y luego utilizar el comando export de la siguiente manera:

```
myvar='I will do it likegeeks'

echo $myvar

export myvar
```

*Ilustración 3. Declarando una variable global*

Sin embargo cuando se declara de esta manera al salir del Shell la variable desaparecerá. Si se desea que sea permanente debe hacerse de la siguiente manera:

```
export myvar='welcome to likegeeks'

likegeeks@likegeeks-VirtualBox ~ $ echo $myvar
welcome to liekgeeks website
likegeeks@likegeeks-VirtualBox ~ $
```

*Ilustración 4. Declarando una variable global permanente*

## Desarrollo (Códigos y ventanas de ejecución).

En la siguiente tabla podemos ver los códigos y las ventanas de ejecución de forma posterior. [14]

Tabla 2. Códigos y su funcionamiento.

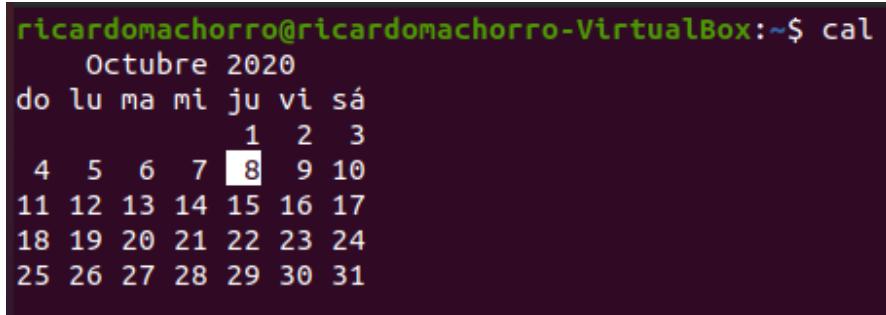
COMANDO	FUNCIONAMIENTO
cal	Muestra un calendario en la salida standard.
clear	Es un comando de linux que borra toda la información del terminal y lo deja como si fuera nuevo.
apt	Es un programa de gestión de paquetes creado por el proyecto Debian.
rm	Es un comando de la familia de sistemas operativos Unix usada para eliminar archivos y directorios del sistema de archivos.
date	Es un comando existente en sistemas Unix y tipo Unix que muestra la hora y la fecha del sistema y el administrador también puede cambiarla.
ifconfig	Es un programa disponible en varias versiones del sistema operativo UNIX, que permite configurar o desplegar numerosos parámetros de las interfaces de red residentes en el núcleo, como la dirección IP (dinámica o estática), o la máscara de red.
exit	Es un comando que sale del shell actual. salida ejecutivo puede pagar el valor de estado de salida especificado
mv	Es un comando de Unix usado para mover o renombrar archivos o directorios del sistema de archivos.
echo	Es un comando para la impresión de un texto en pantalla.
df	Es un comando que nos informa acerca del espacio total, ocupado y libre en nuestro sistema.
ps	Es un comando asociado en el sistema operativo UNIX (estandarizado en POSIX y otros) que permite visualizar el estado de un Proceso (informática).
more	Es un comando para ver (pero no modificar) el contenido de un archivo o comando y visualizarlo por páginas.
time	Es un comando utilizado para determinar el tiempo de ejecución de una operación específica.
du	Es un comando estándar de los sistemas operativos de la familia Unix. Se usa para estimar el uso de espacio en disco duro de un archivo, un directorio en particular o de archivos en un sistema de archivos.

ps -fea	Muestra una instantánea de "todos" los procesos del sistema, los parámetros con los que se levantó el proceso y también los procesos de otros usuarios
less	El comando Less de Linux se usa para mostrar el texto en la pantalla del terminal.
uname	Uname sirve para imprimir información del sistema Linux.
pstree	Es un comando poderoso y útil para mostrar procesos en ejecución en Linux.
man	Es una herramienta de sistemas Unix que se utiliza para documentar y aprender sobre comandos, archivos, llamadas de sistema, etc., en un sistema operativo tal como GNU/Linux.
mkdir	es una orden de los sistemas operativos UNIX, DOS, OS/2 y Microsoft Windows usada para crear un nuevo subdirectorio o carpeta del sistema de archivos
w	Es un programa de Unix que muestra información sobre los usuarios que han iniciado sesión en el sistema y qué están haciendo.
kill -1-9	Fuerza a que un proceso termine
cat	Concatena archivos y los muestra en la salida estándar, es utilizado en el sistema operativo Unix con licencia GNU.
pico	Es un sencillo editor de texto basado en el editor del sistema de mensajes Pine. Al igual que en Pine, los comandos son desplegados en la parte inferior de la pantalla, y se proporciona ayuda sensible al contexto. Conforme
who	Este comando muestra rápidamente los usuarios que actualmente están autenticados en el sistema.
trap -l	
fg	Reanuda trabajos suspendidos poniéndolos en foreground (primer plano) o trabajos en background los pasa a primer plano.
nano	Es un editor de texto para sistemas Unix basado en curses.
bash	Es un programa informático cuya función consiste en interpretar órdenes.
pwd	Se utiliza para imprimir el nombre del directorio actual en una sesión de comandos bajo un sistema operativo Unix o derivado.
cd	Sirve para cambiar la ruta actual de la terminal y ubicarnos en una carpeta o directorio específico.
vi	Vi es un editor de texto que maneja en memoria el texto entero de un archivo. Es el editor clásico de UNIX (se encuentra en todas las versiones).

wc	Es un comando utilizado en el sistema operativo Unix que permite realizar diferentes conteos desde la entrada estándar, ya sea de palabras, caracteres o saltos de líneas.
su	
ls	Es muy útil para ver los archivos y directorios que tenemos dentro del directorio en el que estamos.
apt-get	Instala/informa sobre los paquetes resolviendo las dependencias, los paquetes que instala los consigne de Internet (de /etc/apt/sources.list).
sudo	Que permite a los usuarios ejecutar programas con los privilegios de seguridad de otro usuario (normalmente el usuario root) de manera segura, convirtiéndose así temporalmente en súper usuario.
ls -la	Lista contenido de un directorio, y con l lo hace en formato largo, mientras que con a lo hará para archivos ocultos tambien.

Tabla 2. Lista de comandos

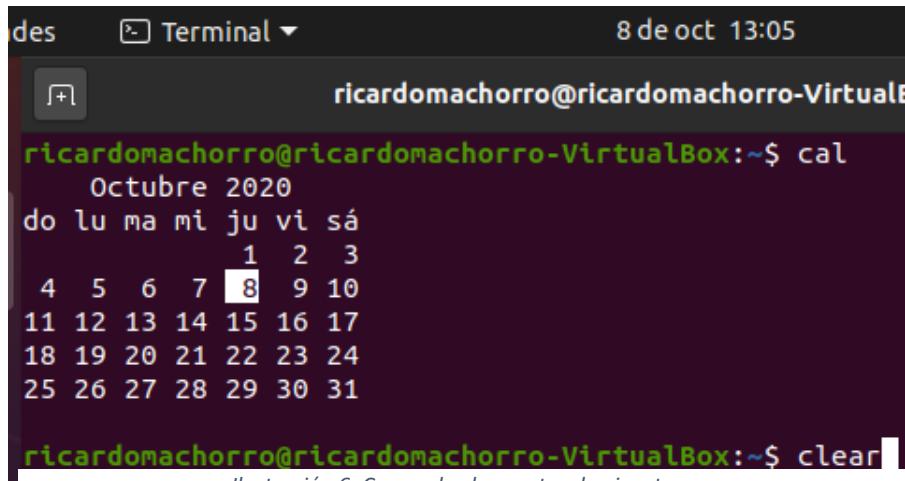
## Comando cal



```
ricardomachorro@ricardomachorro-VirtualBox:~$ cal
          Octubre 2020
do lu ma mi ju vi sa
      1  2  3
 4  5  6  7  8  9 10
11 12 13 14 15 16 17
18 19 20 21 22 23 24
25 26 27 28 29 30 31
```

Ilustración 5. Comando cal

## Comando clear antes de ejecutarse



```
des  [-] Terminal ▾          8 de oct 13:05
ricardomachorro@ricardomachorro-VirtualBox:~$ cal
          Octubre 2020
do lu ma mi ju vi sa
      1  2  3
 4  5  6  7  8  9 10
11 12 13 14 15 16 17
18 19 20 21 22 23 24
25 26 27 28 29 30 31

ricardomachorro@ricardomachorro-VirtualBox:~$ clear
```

Ilustración 6. Comando clear antes de ejecutarse

## Comando clear después de ejecutarse

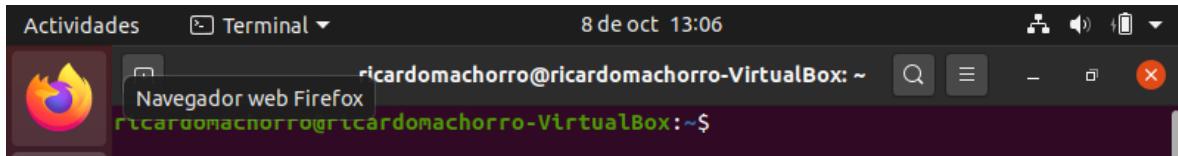


Ilustración 7. Comando clear después de ejecutarse

## Comando apt

```
- RemoveCaches (13: Permiso denegado)
ricardomachorro@ricardomachorro-VirtualBox:~$ sudo apt update
[sudo] contraseña para ricardomachorro:
Obj:1 http://mx.archive.ubuntu.com/ubuntu focal InRelease
Des:2 http://security.ubuntu.com/ubuntu focal-security InRelease [107 kB]
Des:3 http://mx.archive.ubuntu.com/ubuntu focal-updates InRelease [111 kB]
Des:4 http://mx.archive.ubuntu.com/ubuntu focal-backports InRelease [98.3 kB]
Des:5 http://mx.archive.ubuntu.com/ubuntu focal-updates/main i386 Packages [347 kB]
Des:6 http://security.ubuntu.com/ubuntu focal-security/main amd64 DEP-11 Metadata [55.6 kB]
D:p://security.ubuntu.com/ubuntu focal-security/universe amd64 DEP-11 Metadata [55.6 kB]
Des:8 http://mx.archive.ubuntu.com/ubuntu focal-updates/main amd64 Packages [58 8 kB]
Des:9 http://mx.archive.ubuntu.com/ubuntu focal-updates/main amd64 DEP-11 Metadata [209 kB]
Des:10 http://mx.archive.ubuntu.com/ubuntu focal-updates/universe i386 Packages [501 kB]
Des:11 http://mx.archive.ubuntu.com/ubuntu focal-updates/universe amd64 Packages [666 kB]
Des:12 http://mx.archive.ubuntu.com/ubuntu focal-updates/universe amd64 DEP-11 Metadata [200 kB]
Des:13 http://mx.archive.ubuntu.com/ubuntu focal-updates/multiverse amd64 DEP-11 Metadata [2 468 B]
Des:14 http://mx.archive.ubuntu.com/ubuntu focal-backports/universe amd64 DEP-11 Metadata [1 768 B]
Descargados 2 912 kB en 6s (466 kB/s)
```

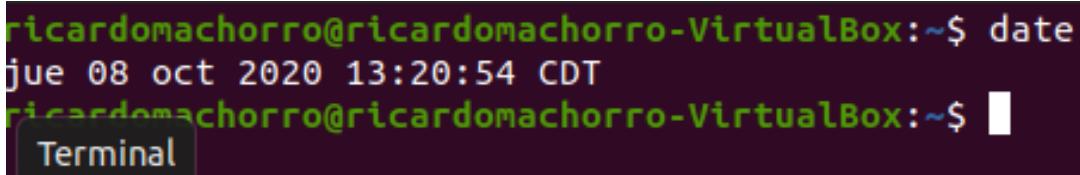
Ilustración 8. Comando apt

## Comando rm

```
ricardomachorro@ricardomachorro-VirtualBox:~$ ls
Descargas Escritorio Imágenes Plantillas Vídeos
Documentos hola Música Público
ricardomachorro@ricardomachorro-VirtualBox:~$ rm hola
ricardomachorro@ricardomachorro-VirtualBox:~$ ls
Descargas Escritorio Música Público
Documentos Imágenes Plantillas Vídeos
ricardomachorro@ricardomachorro-VirtualBox:~$
```

Ilustración 9. Comando rm

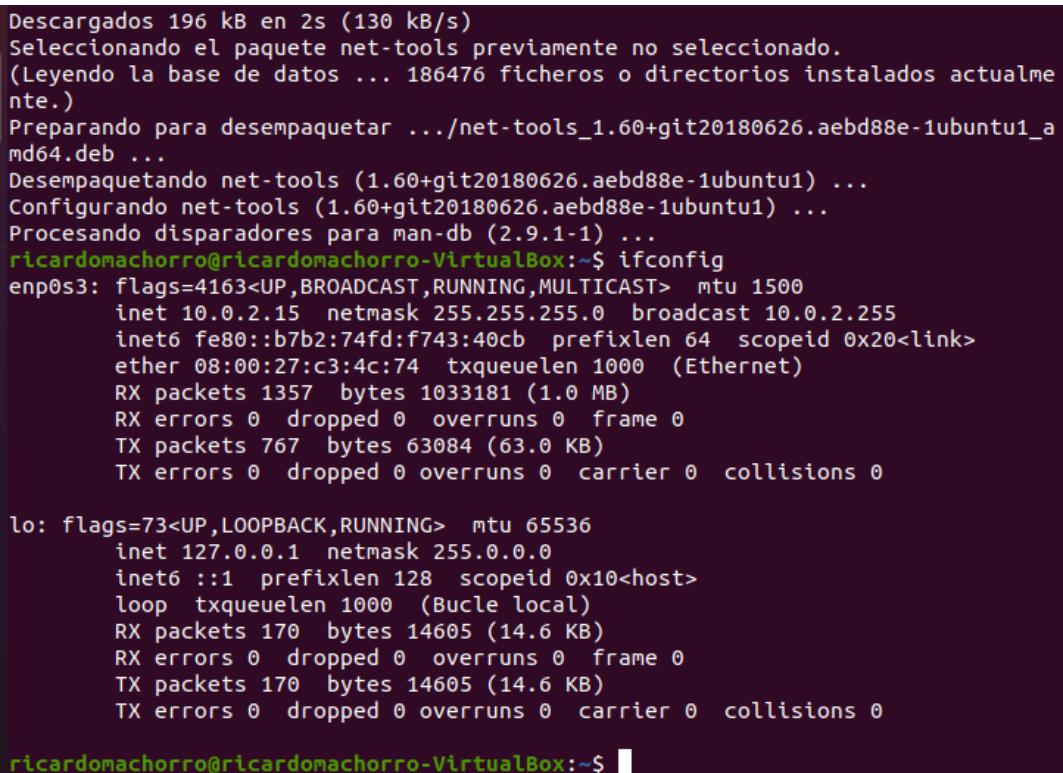
## Comando Date.



```
ricardomachorro@ricardomachorro-VirtualBox:~$ date
jue 08 oct 2020 13:20:54 CDT
ricardomachorro@ricardomachorro-VirtualBox:~$
```

Ilustración 10. Comando date

## Comando ifconfig.



```
Descargados 196 kB en 2s (130 kB/s)
Seleccionando el paquete net-tools previamente no seleccionado.
(Leyendo la base de datos ... 186476 ficheros o directorios instalados actualmente.)
Preparando para desempaquetar .../net-tools_1.60+git20180626.aebd88e-1ubuntu1_amd64.deb ...
Desempaquetando net-tools (1.60+git20180626.aebd88e-1ubuntu1) ...
Configurando net-tools (1.60+git20180626.aebd88e-1ubuntu1) ...
Procesando disparadores para man-db (2.9.1-1) ...
ricardomachorro@ricardomachorro-VirtualBox:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
        inet 10.0.2.15 netmask 255.255.255.0 broadcast 10.0.2.255
        inet6 fe80::b7b2:74fd:f743:40cb prefixlen 64 scopeid 0x20<link>
          ether 08:00:27:c3:4c:74 txqueuelen 1000 (Ethernet)
            RX packets 1357 bytes 1033181 (1.0 MB)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 767 bytes 63084 (63.0 KB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
        inet 127.0.0.1 netmask 255.0.0.0
        inet6 ::1 prefixlen 128 scopeid 0x10<host>
          loop txqueuelen 1000 (Bucle local)
            RX packets 170 bytes 14605 (14.6 KB)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 170 bytes 14605 (14.6 KB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

ricardomachorro@ricardomachorro-VirtualBox:~$
```

Ilustración 11. Comando ifconfig

## Comando exit antes de ejecutarse.



```
ricardomachorro@ricardomachorro-VirtualBox:~$ exit
```

Ilustración 12. Comando exit antes de ejecutarse

## Comando exit después de ejecutarse

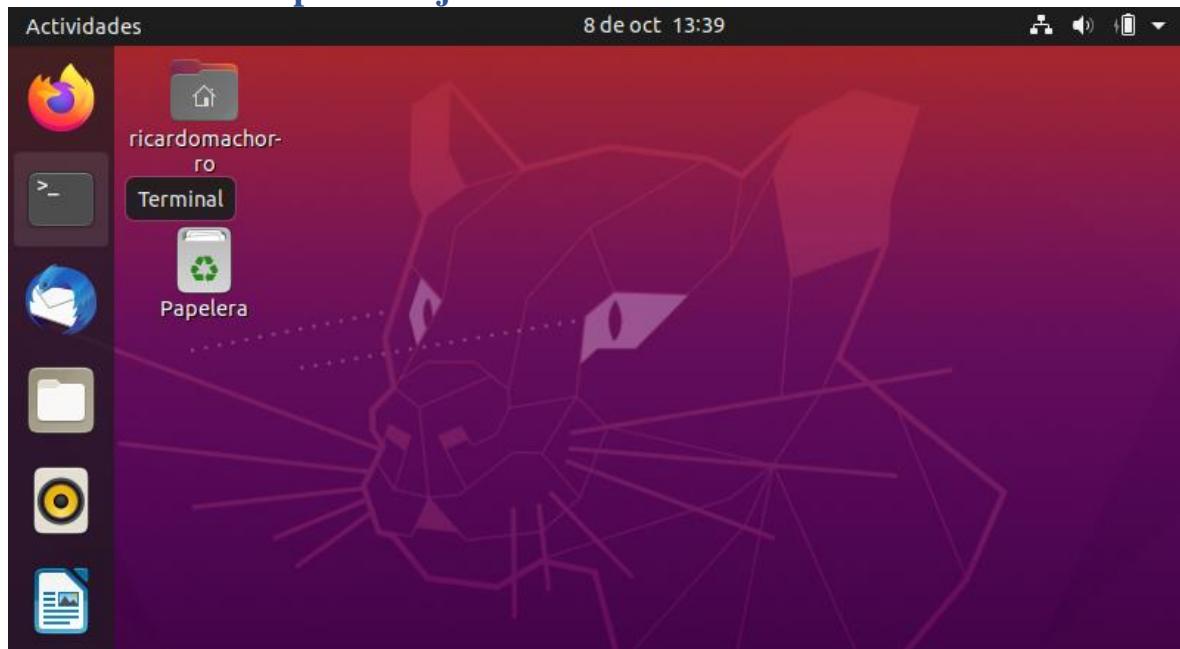


Ilustración 13. Comando exit después de ejecutarse

## Comando mv.

```
ricardomachorro@ricardomachorro-VirtualBox:~$ ls
Descargas Escritorio Música Público
Documentos Imágenes Plantillas Videos
ricardomachorro@ricardomachorro-VirtualBox:~$ touch hola.txt
ricardomachorro@ricardomachorro-VirtualBox:~$ ls
Descargas Escritorio Imágenes Plantillas Videos
Documentos hola.txt Música Público
ricardomachorro@ricardomachorro-VirtualBox:~$ mv hola.txt Documentos/
ricardomachorro@ricardomachorro-VirtualBox:~$ ls
Descargas Escritorio Música Público
Documentos Imágenes Plantillas Videos
ricardomachorro@ricardomachorro-VirtualBox:~$ cd Documentos
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ ls
hola.txt
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ █
```

Ilustración 14. Comando mv

## Comando echo

```
ricardomachorro@ricardomachorro-VirtualBox:~$ ls
Descargas Escritorio Música Público
Documentos Imágenes Plantillas Vídeos
ricardomachorro@ricardomachorro-VirtualBox:~$ touch hola.txt
ricardomachorro@ricardomachorro-VirtualBox:~$ ls
Descargas Escritorio Imágenes Plantillas Vídeos
Documentos hola.txt Música Público
ricardomachorro@ricardomachorro-VirtualBox:~$ mv hola.txt Documentos/
ricardomachorro@ricardomachorro-VirtualBox:~$ ls
Descargas Escritorio Música Público
Documentos Imágenes Plantillas Vídeos
ricardomachorro@ricardomachorro-VirtualBox:~$ cd Documentos
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ ls
hola.txt
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ echo hola
hola
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ █
```

Ilustración 15. Comando echo

## Comando df

```
ricardomachorro@ricardomachorro-VirtualBox:~$ mv hola.txt Documentos/
ricardomachorro@ricardomachorro-VirtualBox:~$ ls
Descargas Escritorio Música Público
Documentos Imágenes Plantillas Vídeos
ricardomachorro@ricardomachorro-VirtualBox:~$ cd Documentos
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ ls
hola.txt
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ echo hola
hola
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ df
S.ficheros   bloques de 1K Usados Disponibles Uso% Montado en
udev          989580      0    989580  0% /dev
tmpfs         203552   1328   202224  1% /run
/dev/sda5     19992176 9635676  9317908 51% /
tmpfs         1017752      0   1017752  0% /dev/shm
tmpfs          5120       4     5116  1% /run/lock
tmpfs         1017752      0   1017752  0% /sys/fs/cgroup
/dev/loop1    261760   261760      0 100% /snap/gnome-3-34-1804/36
/dev/loop2    56704    56704      0 100% /snap/core18/1885
/dev/loop0    56320    56320      0 100% /snap/core18/1880
/dev/loop3    223232  223232      0 100% /snap/gnome-3-34-1804/60
/dev/loop4    63616   63616      0 100% /snap/gtk-common-themes/1
506
/dev/loop5    30720   30720      0 100% /snap/snapd/8542
/dev/loop6    51072   51072      0 100% /snap/snap-store/467
/dev/loop7    31104   31104      0 100% /snap/snapd/9279
/dev/sda1    523248      4   523244  1% /boot/efi
tmpfs        203548     20   203528  1% /run/user/1000
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$
```

Ilustración 16. Comando df

## Comando ps

```
ricardomachorro@ricardomachorro-VirtualBox:~$ cd Documentos
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ ls
hola.txt
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ echo hola
hola
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ df
S. ficheros    bloques de 1K Usados Disponibles Uso% Montado en
udev            989580      0    989580   0% /dev
tmpfs           203552   1328   202224   1% /run
/dev/sda5       19992176 9635676  9317908  51% /
tmpfs           1017752      0   1017752   0% /dev/shm
tmpfs            5120        4     5116   1% /run/lock
tmpfs           1017752      0   1017752   0% /sys/fs/cgroup
/dev/loop1      261760   261760      0 100% /snap/gnome-3-34-1804/36
/dev/loop2      56704    56704      0 100% /snap/core18/1885
/dev/loop0      56320    56320      0 100% /snap/core18/1880
/dev/loop3      223232   223232      0 100% /snap/gnome-3-34-1804/60
/dev/loop4      63616    63616      0 100% /snap/gtk-common-themes/1
506
/dev/loop5      30720    30720      0 100% /snap/snapd/8542
/dev/loop6      51072    51072      0 100% /snap/snap-store/467
/dev/loop7      31104    31104      0 100% /snap/snapd/9279
/dev/sda1       523248      4    523244   1% /boot/efi
tmpfs           203548     20   203528   1% /run/user/1000
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ ps
  PID TTY      TIME CMD
 3326 pts/0    00:00:00 bash
 4908 pts/0    00:00:00 ps
```

Ilustración 17. Comando ps

## Comando more

```
udev            989580      0    989580   0% /dev
tmpfs           203552   1328   202224   1% /run
/dev/sda5       19992176 9635676  9317908  51% /
tmpfs           1017752      0   1017752   0% /dev/shm
tmpfs            5120        4     5116   1% /run/lock
tmpfs           1017752      0   1017752   0% /sys/fs/cgroup
/dev/loop1      261760   261760      0 100% /snap/gnome-3-34-1804/36
/dev/loop2      56704    56704      0 100% /snap/core18/1885
/dev/loop0      56320    56320      0 100% /snap/core18/1880
/dev/loop3      223232   223232      0 100% /snap/gnome-3-34-1804/60
/dev/loop4      63616    63616      0 100% /snap/gtk-common-themes/1
506
/dev/loop5      30720    30720      0 100% /snap/snapd/8542
/dev/loop6      51072    51072      0 100% /snap/snap-store/467
/dev/loop7      31104    31104      0 100% /snap/snapd/9279
/dev/sda1       523248      4    523244   1% /boot/efi
tmpfs           203548     20   203528   1% /run/user/1000
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ ps
  PID TTY      TIME CMD
 3326 pts/0    00:00:00 bash
 4908 pts/0    00:00:00 ps
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ ls
hola.txt
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ more hola.txt
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ more hola.txt
hola
dsds
adsas
```

Ilustración 18. Comando more

## Comando time

```
/dev/loop1      261760  261760          0  100% /snap/gnome-3-34-1804/36
/dev/loop2      56704   56704           0  100% /snap/core18/1885
/dev/loop0      56320   56320           0  100% /snap/core18/1880
/dev/loop3     223232  223232          0  100% /snap/gnome-3-34-1804/60
/dev/loop4      63616   63616           0  100% /snap/gtk-common-themes/1
506
/dev/loop5      30720   30720           0  100% /snap/snapd/8542
/dev/loop6      51072   51072           0  100% /snap/snap-store/467
/dev/loop7      31104   31104           0  100% /snap/snapd/9279
/dev/sda1       523248    4      523244   1% /boot/efi
tmpfs        203548   20            203528   1% /run/user/1000
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ ps
  PID TTY      TIME CMD
 3326 pts/0    00:00:00 bash
 4908 pts/0    00:00:00 ps
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ ls
hola.txt
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ more hola.txt
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ more hola.txt
hola
dsds
adsas
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ time date
jue 08 oct 2020 16:46:50 CDT

real    0m0.001s
user    0m0.001s
sys     0m0.000s
```

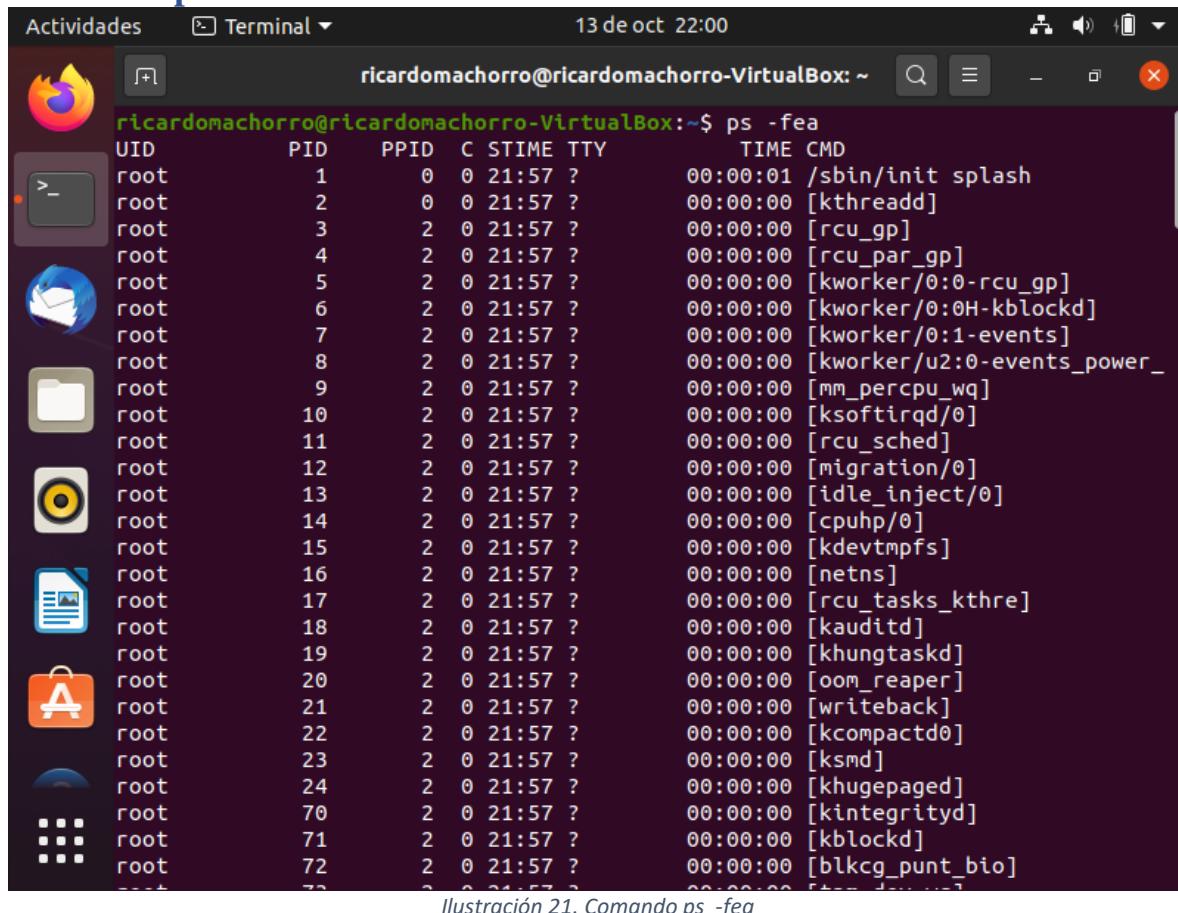
Ilustración 19. Comando time

## Comando du

```
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ du hola.txt
4      hola.txt
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$
```

Ilustración 20. Comando du

## Comando ps -fea

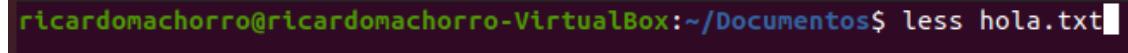


A screenshot of a Linux desktop environment. On the left, there's a dock with icons for a browser, terminal, file manager, and other applications. In the center, a terminal window is open with the command 'ps -fea' entered. The output shows a list of processes, mostly kernel threads, with columns for UID, PID, PPID, C, STIME, TTY, TIME, and CMD.

UID	PID	PPID	C	STIME	TTY	TIME	CMD
root	1	0	0	21:57	?	00:00:01	/sbin/init splash
root	2	0	0	21:57	?	00:00:00	[kthreadd]
root	3	2	0	21:57	?	00:00:00	[rcu_gp]
root	4	2	0	21:57	?	00:00:00	[rcu_par_gp]
root	5	2	0	21:57	?	00:00:00	[kworker/0:0-rcu_gp]
root	6	2	0	21:57	?	00:00:00	[kworker/0:0H-kblockd]
root	7	2	0	21:57	?	00:00:00	[kworker/0:1-events]
root	8	2	0	21:57	?	00:00:00	[kworker/u2:0-events_power_
root	9	2	0	21:57	?	00:00:00	[mm_percpu_wq]
root	10	2	0	21:57	?	00:00:00	[ksoftirqd/0]
root	11	2	0	21:57	?	00:00:00	[rcu_sched]
root	12	2	0	21:57	?	00:00:00	[migration/0]
root	13	2	0	21:57	?	00:00:00	[idle_inject/0]
root	14	2	0	21:57	?	00:00:00	[cpuhp/0]
root	15	2	0	21:57	?	00:00:00	[kdevtmpfs]
root	16	2	0	21:57	?	00:00:00	[netns]
root	17	2	0	21:57	?	00:00:00	[rcu_tasks_kthre]
root	18	2	0	21:57	?	00:00:00	[kaudittd]
root	19	2	0	21:57	?	00:00:00	[khungtaskd]
root	20	2	0	21:57	?	00:00:00	[oom_reaper]
root	21	2	0	21:57	?	00:00:00	[writeback]
root	22	2	0	21:57	?	00:00:00	[kcompactd0]
root	23	2	0	21:57	?	00:00:00	[ksmd]
root	24	2	0	21:57	?	00:00:00	[khugepaged]
root	70	2	0	21:57	?	00:00:00	[kintegrityd]
root	71	2	0	21:57	?	00:00:00	[kblockd]
root	72	2	0	21:57	?	00:00:00	[blkcg_punt_bio]

Ilustración 21. Comando ps -fea

## Comando less antes de ejecutarse



A screenshot of a terminal window. The command 'less hola.txt' is entered. The text 'hola' is visible at the top of the screen.

Ilustración 22. Comando less antes de ejecutarse

## Comando less después de ejecutarse

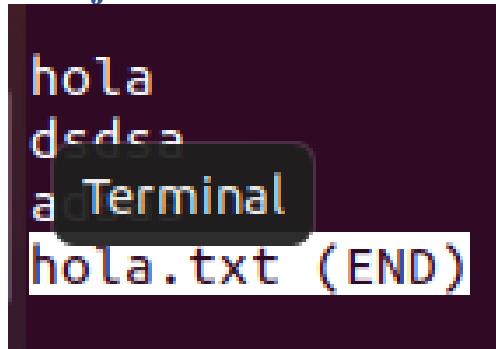


Ilustración 23. Comando less después de ejecutarse

## Función uname

```
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ less hola.txt
[3]+  Detenido                  less hola.txt
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ uname
Linux
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ uname -a
Linux ricardomachorro-VirtualBox 5.4.0-48-generic #52-Ubuntu SMP Thu Sep 10 10:58:49 UTC 2020 x86_64 x86_64 x86_64 GNU/Linux
```

Ilustración 24. Comando uname

## Comando pstree

```
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ less hola.txt
[3]+  Detenido                  less hola.txt
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ uname
Linux
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ uname -a
Linux ricardomachorro-VirtualBox 5.4.0-48-generic #52-Ubuntu SMP Thu Sep 10 10:58:49 UTC 2020 x86_64 x86_64 x86_64 GNU/Linux
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ pstree
systemd—ModemManager—2*[{ModemManager}]
  └── NetworkManager—2*[{NetworkManager}]
    ├── accounts-daemon—2*[{accounts-daemon}]
    ├── acpid
    ├── avahi-daemon—avahi-daemon
    ├── colord—2*[{colord}]
    ├── cron
    ├── cups-browsed—2*[{cups-browsed}]
    ├── cupsd
    ├── dbus-daemon
    ├── gdm3—gdm-session-wor—gdm-x-session—Xorg—{Xorg}
      └── gnome-session-b—ssh-agent
        └── 2*[{gnome+}]
          └── 2*[{gdm-x-session}]
    └── 2*[{gdm3}]
    └── gnome-keyring-d—3*[{gnome-keyring-d}]
    └── ibus-daemon—ibus-dconf—3*[{ibus-dconf}]
      ├── ibus-engine-sim—2*[{ibus-engine-sim}]
      └── ibus-extension—3*[{ibus-extension-}]
```

Ilustración 25. Comando pstree

## Comando man

```
FREE(1)                               User Commands                         FREE(1)

NAME
    free - Display amount of free and used memory in the system

SYNOPSIS
    free [options]

DESCRIPTION
    free displays the total amount of free and used physical and swap mem-
    ory in the system, as well as the buffers and caches used by the ker-
    nel. The information is gathered by parsing /proc/meminfo. The dis-
    played columns are:

        total Total installed memory (MemTotal and SwapTotal in /proc/mem-
               info)

        used Used memory (calculated as total - free - buffers - cache)

        free Unused memory (MemFree and SwapFree in /proc/meminfo)

        shared Memory used (mostly) by tmpfs (Shmem in /proc/meminfo)

        buffers
            Memory used by kernel buffers (Buffers in /proc/meminfo)

        cache Memory used by the page cache and slabs (Cached and SRe-
               claimable in /proc/meminfo)
Manual page free(1) line 1 (press h for help or q to quit)
```

Ilustración 26. Comando man

## Comando mkdir

```
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ mkdir Prueba
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ ls
hola.txt  Prueba
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$
```

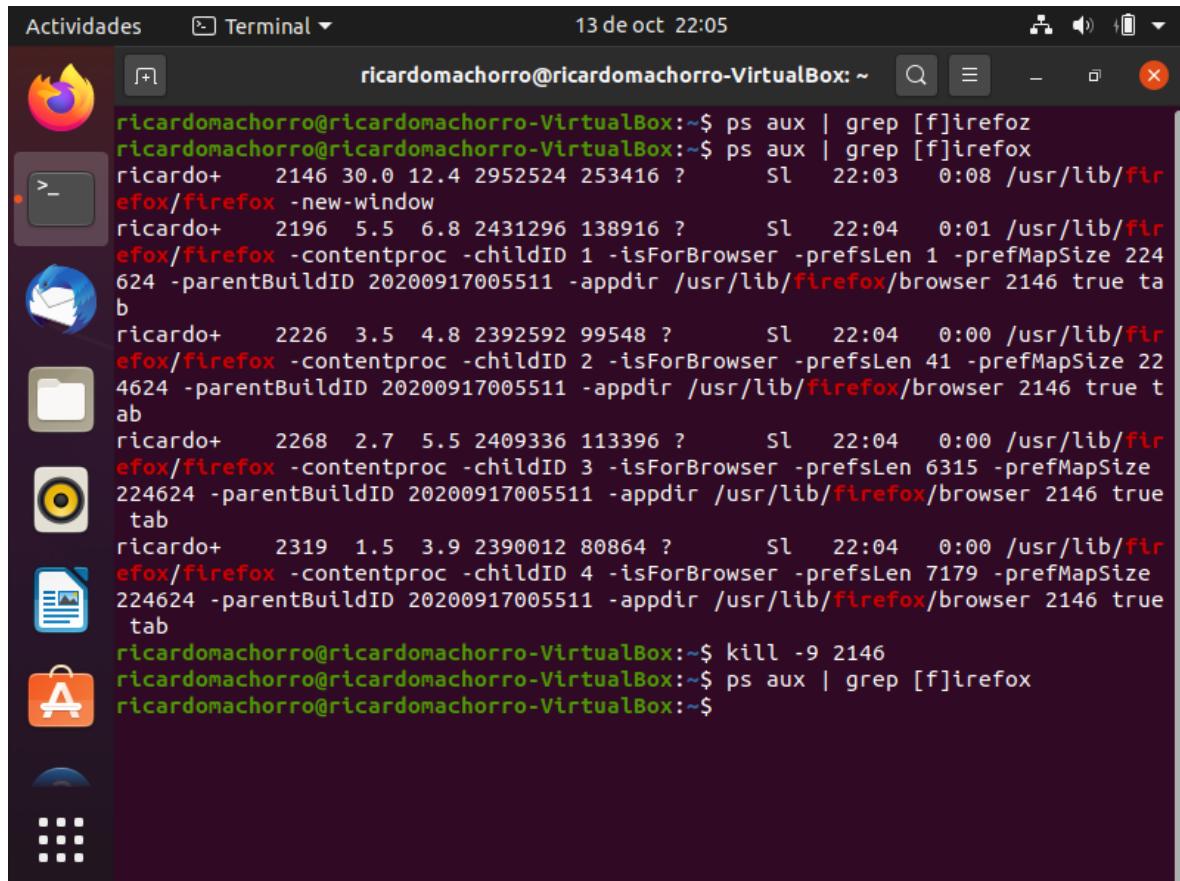
Ilustración 27. Comando mkdir

## Comando w

```
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ w
18:17:00 up  5:16,  1 user,  load average: 0.00, 0.01, 0.00
USUARIO   TTY      DE          LOGIN@    IDLE    JCPU   PCPU WHAT
ricardom :0      :0          13:00    ?xdm?    1:35   0.02s /usr/lib/gdm3/
```

Ilustración 28. Comando w

## Comando Kill -9

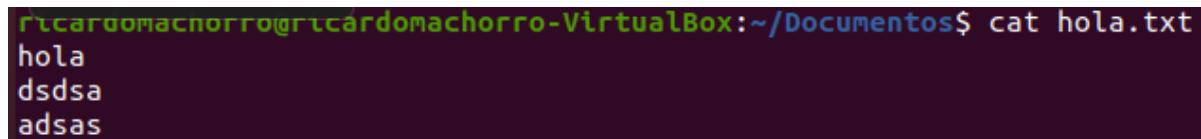


The screenshot shows a Linux desktop environment with a terminal window open. The terminal window title is 'Terminal'. The date and time at the top right are '13 de oct 22:05'. The terminal content is as follows:

```
ricardomachorro@ricardomachorro-VirtualBox:~$ ps aux | grep [f]irefox
ricardomachorro@ricardomachorro-VirtualBox:~$ ps aux | grep [f]irefox
ricardo+ 2146 30.0 12.4 2952524 253416 ? Sl 22:03 0:08 /usr/lib/firefox/firefox -new-window
ricardo+ 2196 5.5 6.8 2431296 138916 ? Sl 22:04 0:01 /usr/lib/firefox/firefox -contentproc -childID 1 -isForBrowser -prefsLen 1 -prefMapSize 224
624 -parentBuildID 20200917005511 -appdir /usr/lib/firefox/browser 2146 true tab
ricardo+ 2226 3.5 4.8 2392592 99548 ? Sl 22:04 0:00 /usr/lib/firefox/firefox -contentproc -childID 2 -isForBrowser -prefsLen 41 -prefMapSize 22
4624 -parentBuildID 20200917005511 -appdir /usr/lib/firefox/browser 2146 true tab
ricardo+ 2268 2.7 5.5 2409336 113396 ? Sl 22:04 0:00 /usr/lib/firefox/firefox -contentproc -childID 3 -isForBrowser -prefsLen 6315 -prefMapSize 22
4624 -parentBuildID 20200917005511 -appdir /usr/lib/firefox/browser 2146 true tab
ricardo+ 2319 1.5 3.9 2390012 80864 ? Sl 22:04 0:00 /usr/lib/firefox/firefox -contentproc -childID 4 -isForBrowser -prefsLen 7179 -prefMapSize 22
4624 -parentBuildID 20200917005511 -appdir /usr/lib/firefox/browser 2146 true tab
ricardomachorro@ricardomachorro-VirtualBox:~$ kill -9 2146
ricardomachorro@ricardomachorro-VirtualBox:~$ ps aux | grep [f]irefox
ricardomachorro@ricardomachorro-VirtualBox:~$
```

Ilustración 29. Comando kill -9

## Función cat



```
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ cat hola.txt
hola
dsds
adsas
```

Ilustración 30. Función cat

## Comando pico antes de compilarse



```
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ pico hola.txt
```

Ilustración 31. Comando pico antes de compilarse

## Comando pico después de compilarse

GNU nano 4.8 hola.txt

```
hola
dsds
adsas
```

Ilustración 32. Comando pico después de compilarse

## Comando who

```
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ pico hola.txt
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ who
ricardomachorro :0          2020-10-08 13:00 (:0)
r Terminal |chorro@ricardomachorro-VirtualBox:~/Documentos$
```

Ilustración 33. Comando who

## Comando fg

```
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ fg
less hola.txt
[3]+  Detenido                  less hola.txt
```

Ilustración 34. Comando fg

## Comando nano antes de ejecutarse

```
Navegador Web Firefox
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ nano hola.txt
```

Ilustración 35. Comando nano antes de ejecutarse

## Comando nano después de ejecutarse

Navegador Web Firefox

GNU nano 4.8 hola.txt

```
hola
dsds
adsas
```

Ilustración 36. Comando nano después de ejecutarse

## Comando pwd

```
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ nano hola.txt
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ pwd
/home/ricardomachorro/Documentos
```

Ilustración 37. Comando pwd

## Comando cd

```
ricardomachorro@ricardomachorro-VirtualBox:~$ cd Documentos
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$
```

Ilustración 38. Comando cd

## Comando vi antes de ejecutarse

```
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ vi hola.txt
```

*Ilustración 39. Comando vi antes de ejecutarse*

## Comando vi luego de ejecutarse

*Ilustración 40. Comando vi después de ejecutarse*

## Comando wc

```
ricardomachorro@ricardomachorro-VirtualBox:~$ cd Documentos  
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ wc hola.txt  
3 3 18 hola.txt
```

Ilustración 41. Comando wc

## Comando ls

```
ricardomachorro@ricardomachorro-VirtualBox:~$ ls
Descargas    Escritorio   Música      Público
Documentos   Imágenes     Plantillas  Vídeos
```

Ilustración 42. Comando ls

## Comando apt-get

```
ricardomachorro@ricardomachorro-VirtualBox:~$ ls
Descargas  Escritorio  Música  Públco
Documentos  Imágenes  Plantillas  Vídeos
ricardomachorro@ricardomachorro-VirtualBox:~$ apt-get moo
          (_)
          (oo)
         /-----\
        /  |  || |
       *  \---/ \
          ~~  ~~
... "Have you mooed today?" ...
```

Ilustración 43. Comando apt-get

## Comando sudo

```
ricardomachorro@ricardomachorro-VirtualBox:~$ sudo /sbin/ifconfig
[sudo] contraseña para ricardomachorro:
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
      inet 10.0.2.15  netmask 255.255.255.0  broadcast 10.0.2.255
      inet6 fe80::b7b2:74fd:40cb  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:c3:4c:74  txqueuelen 1000  (Ethernet)
      RX packets 32239  bytes 25860266 (25.8 MB)
      RX errors 0  dropped 0  overruns 0  frame 0
      TX packets 15876  bytes 1004672 (1.0 MB)
      TX errors 0  dropped 0  overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
      inet 127.0.0.1  netmask 255.0.0.0
      inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Bucle local)
      RX packets 401  bytes 35114 (35.1 KB)
      RX errors 0  dropped 0  overruns 0  frame 0
      TX packets 401  bytes 35114 (35.1 KB)
      TX errors 0  dropped 0  overruns 0  carrier 0  collisions 0
```

Ilustración 44. Comando sudo

## Direccionamiento absoluto y relativo (3 ejemplos por tipo explicados de forma conjunta).

Debemos aclarar que las rutas relativas indican el camino para encontrar un elemento, pero basándonos en el directorio desde el que se ejecuta la orden, desde el directorio en donde nos encontramos posicionados. En el caso de la absoluta se indica toda la ruta del archivo incluyendo el directorio raíz.

### Ejemplo 1 (ruta absoluta y relativa)

Podemos ver que al encontrarnos en nuestra ruta accedemos al home y luego a Ubuntu de forma relativa, ya que posteriormente desde esa misma ubicación entramos a nuestro escritorio.

```
ubuntu@ubuntu:/$ cd home/ubuntu  
ubuntu@ubuntu:~$ cd Desktop  
ubuntu@ubuntu:~/Desktop$ █
```

Ilustración 45. Ruta relativa al escritorio

Ahora bien, de forma absoluta lo podemos hacer en una sola línea. Esta manera no sería conveniente si tuviéramos que acceder a un conjunto de carpetas donde la ruta es demasiados larga, para efectos de rutas muy largas será mejor emplear las relativas.

```
ubuntu@ubuntu:/$ cd /home/ubuntu/Desktop  
ubuntu@ubuntu:~/Desktop$ █
```

Ilustración 46. Ruta absoluta al escritorio

### Ejemplo 2 (ruta absoluta y relativa)

Supongamos que tenemos las 3 carpetas conteniendo a un archivo de tipo texto.



Ilustración 47. Carpetas

La manera de acceder absoluta seria tomando en cuenta todo el directorio, en la parte de abajo se muestra como estamos accediendo desde home hasta llegar a la carpeta 3, con lo que estamos considerando todo desde la ruta “Raiz”

```
ubuntu@ubuntu:/$ cd /home/ubuntu/Desktop/carpeta1/carpeta2/carpeta3  
ubuntu@ubuntu:~/Desktop/carpeta1/carpeta2/carpeta3$
```

Ilustración 48. Ruta absoluta a las carpetas

Ahora bien, supongamos que ya hemos accedido a la carpeta 1, por medio de rutas relativas podemos ahorrarnos volver a escribir toda la dirección de la ruta y solamente partir desde la carpeta 1 en la cual nos encontramos.

```
ubuntu@ubuntu:~/Desktop/carpeta1$ cd carpeta2/carpeta3  
ubuntu@ubuntu:~/Desktop/carpeta1/carpeta2/carpeta3$
```

Ilustración 49. Rutas relativas a las carpetas

### Ejemplo 3 (ruta absoluta y relativa)

Hemos creado 10 carpetas, una dentro de la otra.

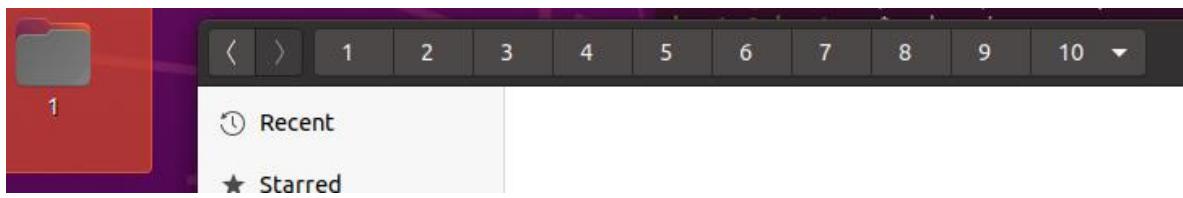


Ilustración 50. Creación de carpetas

Verificamos con ls hacia donde queremos movernos en la ruta.

```
ubuntu@ubuntu:/$ ls  
bin  cdrom  etc  lib   lib64  media  opt   rofs  run   snap  sys  usr  
boot dev    home  lib32  libx32  mnt    proc   root  sbin  srv   tmp  var  
ubuntu@ubuntu:/$
```

Ilustración 51. Ver las carpetas

De la manera absoluta podemos desplazarnos hasta la carpeta 5

```
ubuntu@ubuntu:/$ cd /home/ubuntu/Desktop/1/2/3/4/5  
ubuntu@ubuntu:~/Desktop/1/2/3/4/5$
```

Ilustración 52. Ruta absoluta a la carpeta 5

Ahora bien, si quisieramos llegar al 10 podemos mejor usar una ruta relativa para no tener que volver a escribir todo de nuevo y poder llegar más rápido.

```
ubuntu@ubuntu:~/Desktop/1/2/3/4/5$ cd 6/7/8/9/10  
ubuntu@ubuntu:~/Desktop/1/2/3/4/5/6/7/8/9/10$ █
```

Ilustración 53. Ruta relativa a la carpeta 10

Ahora bien podemos retroceder al nivel cinco y luego subir al nueve, todo partiendo desde donde nos encontrábamos.

```
ubuntu@ubuntu:~/Desktop/1/2/3/4/5/6/7/8/9/10$ cd ../../../../../../6/7/8/9  
ubuntu@ubuntu:~/Desktop/1/2/3/4/5/6/7/8/9$ █
```

Ilustración 54. Ruta relativa a la carpeta 9

## Borrado de archivos “rm y comodines \*?”

Ejemplo 1 Borrado con “rm” de un archivo dentro de 3 carpetas.

Procedemos a buscar el archivo en su ruta y dentro de la misma podemos eliminar el archivo de texto como se muestra en la consola.

Aquí esta una imagen del antes.



Ilustración 55. Carpetas

Nuestra carpeta queda vacía después del borrado.

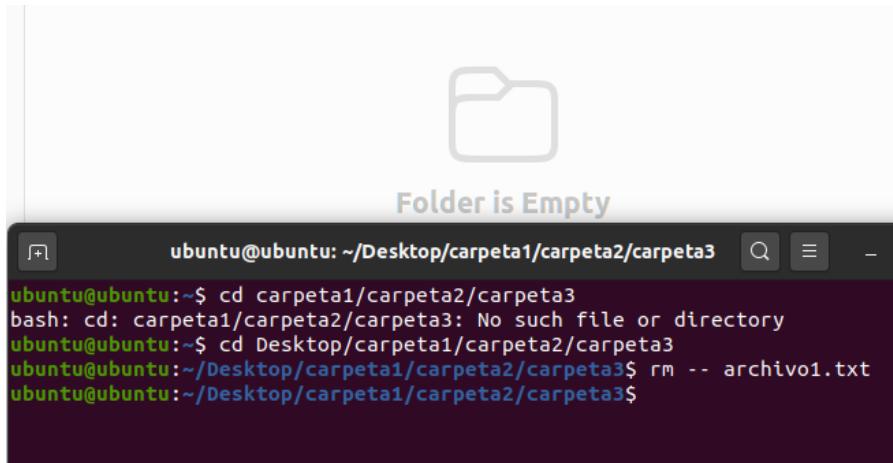


Ilustración 56. Borrado con rm

#### Ejemplo 2 Borrado con “rm” con el uso del comodín \*.

Ahora tenemos 3 archivos en nuestra carpeta3, con el uso del comodín podemos eliminar todo lo que se encuentra dentro de la misma de un solo golpe.

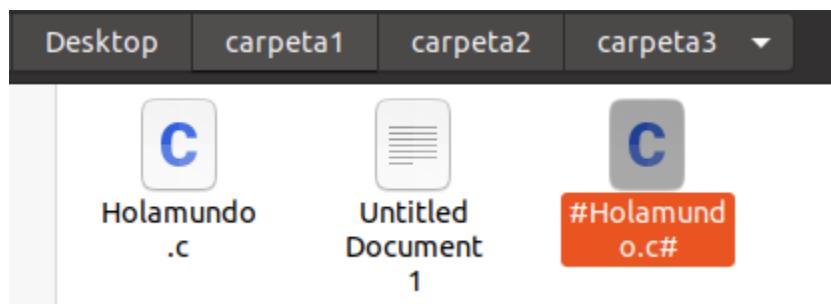


Ilustración 57. Archivos de la carpeta 3

```
ubuntu@ubuntu:~/Desktop/carpeta1/carpeta2/carpeta3$ rm *
```

Ilustración 58. Borrado con comodín\*

#### Ejemplo 3 Borrado con “rm” con el uso del comodín ?.

Aquí lo único que hacemos es usar el ? como un carácter que nos permitirá sustituir ese espacio en la palabra por cualquier letra, como se muestra en ejemplo todos y cada uno de los archivos debieron ser eliminados.



Ilustración 59. Archivos con nombre similar

```
ubuntu@ubuntu:~/Desktop/carpeta1/carpeta2/carpeta3$ rm Hol?.c
ubuntu@ubuntu:~/Desktop/carpeta1/carpeta2/carpeta3$
```

Ilustración 60. Borrado con rm usando comodín?

#### Ejemplo 4 Borrado con “rm”.

Tenemos los siguientes archivos y solo deseamos eliminar los que terminan en s, por lo que aplicamos el comodín ? para poder eliminar los elementos sin importar que los primeros 3 varíen.



Ilustración 61. Archivos que terminan en la misma letra

```
ubuntu@ubuntu:~/Desktop/carpeta1/carpeta2/carpeta3$ rm ???s
```

Ilustración 62. Borrado con rm usando ???

### Redireccionamiento utilizando ‘>’ (crea archivo, sobrescribe si existe).

#### Ejemplo 1

Primero creamos tres archivos donde guardamos la fecha, después con el comando cat mostramos el contenido de uno de estos archivos.

Con el comando ls mostramos los archivos que tenemos en el equipo, usamos ‘>’ para guardar el resultado del comando ls en un archivo de texto llamado salidad.txt

Después mostramos los archivos que tenemos en el equipo, y vemos que ya ha sido creado salidad.txt. Por último con el comando cat mostramos lo que contiene este archivo, y notamos que se contiene a sí mismo, esto sucede porque ‘>’ crea los archivos antes de realizar el redireccionamiento, así que cuando realizo el redireccionamiento el archivo salidad.txt ya había sido creado y también se guardó.

```
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ulises@UlisesJE:~$ date>fecha1
ulises@UlisesJE:~$ date>fecha2
ulises@UlisesJE:~$ date>fecha3
ulises@UlisesJE:~$ cat fecha1
dom 11 oct 2020 19:32:26 CDT
ulises@UlisesJE:~$ ls
Descargas Escritorio fecha2 hola Música Público
Documentos fecha1 fecha3 Imágenes Plantillas Videos
ulises@UlisesJE:~$ ls>salidad.txt
ulises@UlisesJE:~$ ls
Descargas Escritorio fecha2 hola Música Público Videos
Documentos fecha1 fecha3 Imágenes Plantillas salida.txt
```

Ilustración 63. Redireccionamiento > de un comando a un archivo

```
ulises@UlisesJE:~$ cat salida.txt
Descargas
Documentos
Escritorio
fecha1
fecha2
fecha3
hola
Imágenes
Música
Plantillas
Público
salidad.txt
Videos
ulises@UlisesJE:~$ █
```

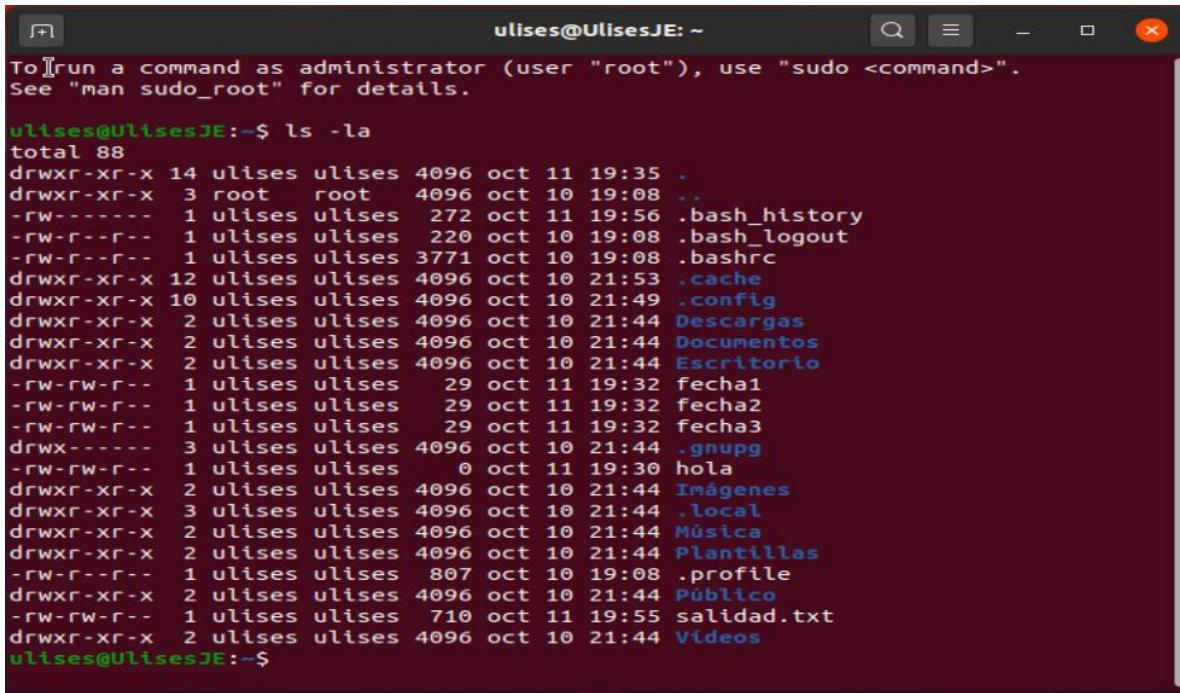
Ilustración 64. Mostrando el contenido del archivo

## Ejemplo 2

En este ejemplo, se ejecuta un comando, posteriormente se redirecciona el resultado de ese comando a un archivo llamado valga la redundancia archivo.txt así que ya no se ve el resultado del comando en la terminal, sino que se guarda en ese archivo.

```
ulises@UlisesJE:~$ ls -la>archivo.txt
ulises@UlisesJE:~$
```

Ilustración 65. Redireccionamiento > de otro comando a un archivo



```
ulises@UlisesJE:~$ ls -la
total 88
drwxr-xr-x 14 ulises ulises 4096 oct 11 19:35 .
drwxr-xr-x  3 root   root   4096 oct 10 19:08 ..
-rw-----  1 ulises ulises  272 oct 11 19:56 .bash_history
-rw-r--r--  1 ulises ulises  220 oct 10 19:08 .bash_logout
-rw-r--r--  1 ulises ulises 3771 oct 10 19:08 .bashrc
drwxr-xr-x 12 ulises ulises 4096 oct 10 21:53 .cache
drwxr-xr-x 10 ulises ulises 4096 oct 10 21:49 .config
drwxr-xr-x  2 ulises ulises 4096 oct 10 21:44 Descargas
drwxr-xr-x  2 ulises ulises 4096 oct 10 21:44 Documentos
drwxr-xr-x  2 ulises ulises 4096 oct 10 21:44 Escritorio
-rw-rw-r--  1 ulises ulises  29 oct 11 19:32 fecha1
-rw-rw-r--  1 ulises ulises  29 oct 11 19:32 fecha2
-rw-rw-r--  1 ulises ulises  29 oct 11 19:32 fecha3
drwx----- 3 ulises ulises 4096 oct 10 21:44 .gnupg
-rw-rw-r--  1 ulises ulises  0 oct 11 19:30 hola
drwxr-xr-x  2 ulises ulises 4096 oct 10 21:44 Imágenes
drwxr-xr-x  3 ulises ulises 4096 oct 10 21:44 .local
drwxr-xr-x  2 ulises ulises 4096 oct 10 21:44 Música
drwxr-xr-x  2 ulises ulises 4096 oct 10 21:44 Plantillas
-rw-r--r--  1 ulises ulises  807 oct 10 19:08 .profile
drwxr-xr-x  2 ulises ulises 4096 oct 10 21:44 Público
-rw-rw-r--  1 ulises ulises 710 oct 11 19:55 salida.txt
drwxr-xr-x  2 ulises ulises 4096 oct 10 21:44 Videos
ulises@UlisesJE:~$
```

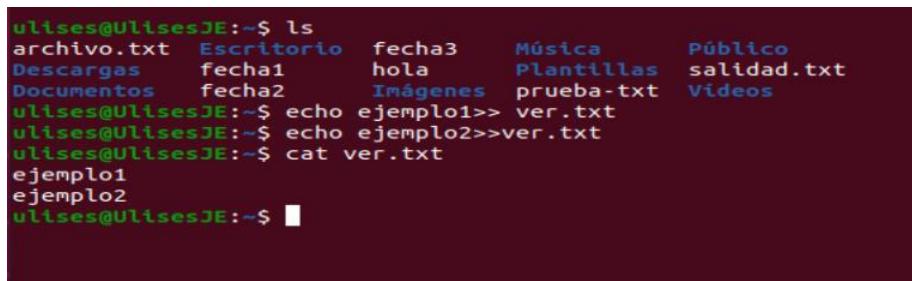
Ilustración 66. Ejecución del comando sin redireccionar

## Redireccionamiento usando ‘>>’ (crea si no existe, agrega al archivo existente).

### Ejemplo 1

Se realiza una impresión de las palabras ejemplo1 y ejemplo2. Cuando se imprimió ejemplo 1 se creó el archivo ver.txt, del mismo modo con ejemplo2, sin embargo para redireccionar se usó ‘>>’ lo que permitió que ejemplo2 se agregara en el archivo y no se sobrescribiera.

Al momento de ver su contenido se puede observar que efectivamente ambas palabras están en el archivo.



```
ulises@UlisesJE:~$ ls
archivo.txt Escritorio fecha3 Música Público
Descargas   fecha1   hola   Plantillas salida.txt
Documentos  fecha2   Imágenes prueba-txt Videos
ulises@UlisesJE:~$ echo ejemplo1>> ver.txt
ulises@UlisesJE:~$ echo ejemplo2>>ver.txt
ulises@UlisesJE:~$ cat ver.txt
ejemplo1
ejemplo2
ulises@UlisesJE:~$
```

Ilustración 67. Redireccionamiento usando >>

## Ejemplo 2

Tenemos un archivo llamado ‘fecha 3’ el cual contenía una instantánea de una fecha determinada, ejecutamos un comando y lo redireccionamos a ‘fecha 3’, al momento de ver que contiene el archivo ‘fecha 3’, se observa que efectivamente se añadió el resultado de ese comando y no se sobrescribió.

```
ulises@UlisesJE:~$ cat fecha3
dom 11 oct 2020 19:32:51 CDT
ulises@UlisesJE:~$ ps -ef>>fecha3
ulises@UlisesJE:~$ cat fecha3
dom 11 oct 2020 19:32:51 CDT
UID      PID  C STIME TTY          TIME CMD
root      1  0 19:17 ?
root      2  0 19:17 ?
root      3  2 19:17 ?
root      4  2 19:17 ?
root      6  2 19:17 ?
root      7  2 19:17 ?
root      9  2 19:17 ?
root     10  2 19:17 ?
root     11  2 19:17 ?
root     12  2 19:17 ?
root     13  2 19:17 ?
root     14  2 19:17 ?
root     15  2 19:17 ?
root     16  2 19:17 ?
root     17  2 19:17 ?
root     18  2 19:17 ?
root     19  2 19:17 ?
```

Ilustración 68. Redireccionamiento de otro comando usando >>

## Instalar en línea de comandos del paquete gimp

Se escribe los comandos:

- sudo add-apt-repository ppa:otto-kesselgulasch/gimp
- sudo apt update
- sudo apt install gimp

Para que así se descargue la versión oficial de gimp de su repositorio.

```
ricardomachorro@ricardomachorro-VirtualBox:~$ sudo add-apt-repository ppa:otto-
kesselgulasch/gimp
Hi,
for personal reasons my PPA activities are dropped for the time being.
Sorry
Thorsten
This PPA is for Ubuntu >=18.04 and Linux Mint derivates
Installing:
open a terminal and type:
sudo add-apt-repository ppa:otto-kesselgulasch/gimp
sudo apt-get update
sudo apt-get install gimp

Removing:
open a terminal and type:
sudo apt-get install ppa-purge
sudo ppa-purge ppa:otto-kesselgulasch/gimp
```

Ilustración 69. Instalar en línea de comandos del paquete gimp

```

1 Metadata [1 768 B]
Descargados 4 446 kB en 15s (292 kB/s)
Leyendo lista de paquetes... Hecho
ricardomachorro@ricardomachorro-VirtualBox:~$ sudo apt update
Obj:1 http://mx.archive.ubuntu.com/ubuntu focal InRelease
Obj:2 http://ppa.launchpad.net/otto-kesselgulasch/gimp/ubuntu focal InRelease
Obj:3 http://mx.archive.ubuntu.com/ubuntu focal-updates InRelease
Obj:4 http://security.ubuntu.com/ubuntu focal-security InRelease
Obj:5 http://mx.archive.ubuntu.com/ubuntu focal-backports InRelease
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Se pueden actualizar 127 paquetes. Ejecute «apt list --upgradable» para verlos.
ricardomachorro@ricardomachorro-VirtualBox:~$ sudo apt install gimp
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
No se pudieron instalar algunos paquetes. Esto puede significar que
usted pidió una situación imposible o, si está usando la distribución
inestable, que algunos paquetes necesarios aún no se han creado o se
han sacado de «Incoming».
La siguiente información puede ayudar a resolver la situación:

Los siguientes paquetes tienen dependencias incumplidas:
  gimp : Depende: libgimp2.0 (>= 2.10.18) pero no va a instalarse
          Depende: libgimp2.0 (<= 2.10.18-z) pero no va a instalarse
          Depende: libgegl-0.4-0 (>= 0.4.22) pero no va a instalarse
E: No se pudieron corregir los problemas, usted ha retenido paquetes rotos.
ricardomachorro@ricardomachorro-VirtualBox:~$ 
```

Ilustración 70. Comando para instalar gimp

## Instalar slack por el entorno gráfico.

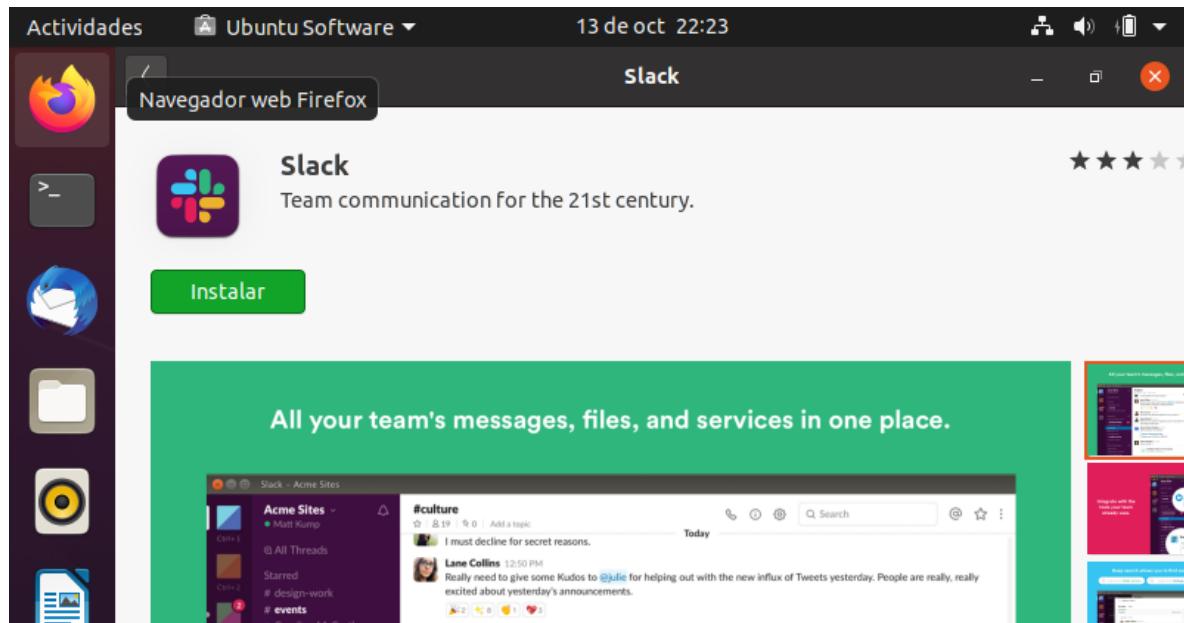


Ilustración 71. Instalar slack

## Jerarquía de directorios

Realizar al menos 5 ejemplos utilizando la jerarquía de directorios para mover y copiar archivos y directorios desde terminal.

### Ejemplo 1.

Para el primer ejercicio realizamos el movimiento de una carpeta del directorio Escritorio al de Documentos. Lo primero que realizamos fue movernos del directorio del usuario al de escritorio, esto lo realizamos con el siguiente comando: cd /home/victor/Escritorio, aquí hacemos uso de redirecciónamiento absoluto.

Cambio del directorio de usuario a Escritorio. Una vez en escritorio listamos los elementos, notamos que hay dos carpetas y un programa. Capturas antes de ejecutar el comando.

```
victor@victor-VirtualBox:~$ cd /home/victor/Escritorio
victor@victor-VirtualBox:~/Escritorio$ ls
carpeta  compartir  holamundo.c
victor@victor-VirtualBox:~/Escritorio$
```

Ilustración 72. Ejemplo1 antes de ejecutar el comando- consola

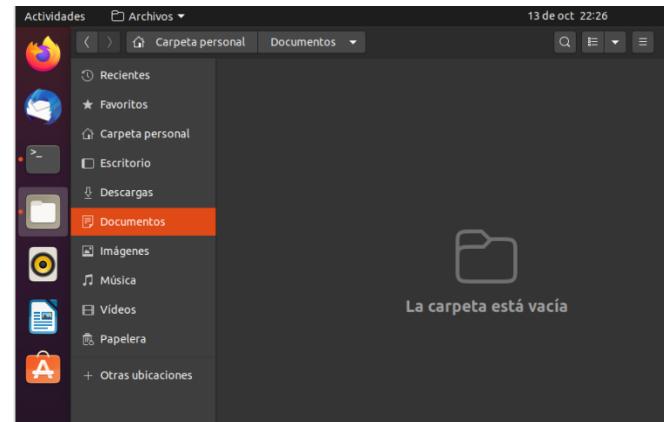


Ilustración 73. Ejemplo1 antes de ejecutar el comando-carpeta

Una vez ejecutado el programa, la carpeta que estaba en el escritorio pasó al de documentos

```
victor@victor-VirtualBox:~$ cd /home/victor/Escritorio
victor@victor-VirtualBox:~/Escritorio$ ls
carpeta  compartir  holamundo.c
victor@victor-VirtualBox:~/Escritorio$ mv carpeta /home/victor/Documentos
victor@victor-VirtualBox:~/Escritorio$
```

Ilustración 74. Ejemplo1 comando ejecutado-consola

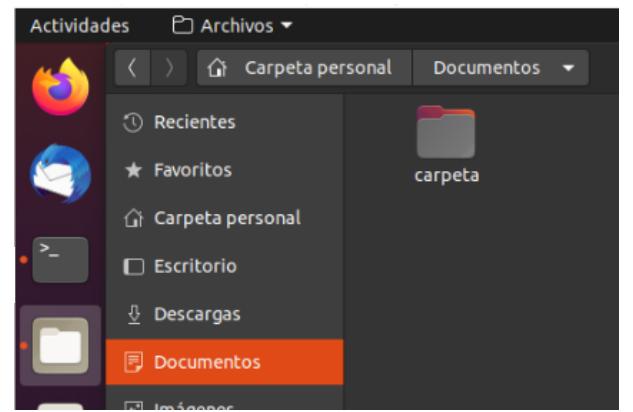


Ilustración 75. Ejemplo1 comando ejecutado-carpeta

## Ejemplo 2

Ahora realizaremos el movimiento de un archivo del directorio Escritorio al de Documentos, para eso el comando que ejecutaremos es: mv holamundo.c /home/victor/Documentos. Capturas antes de ejecutar el comando.

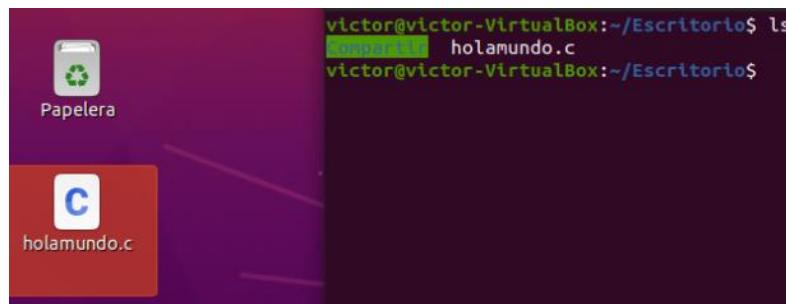


Ilustración 76. Ejemplo2 antes de ejecutar el comando- consola

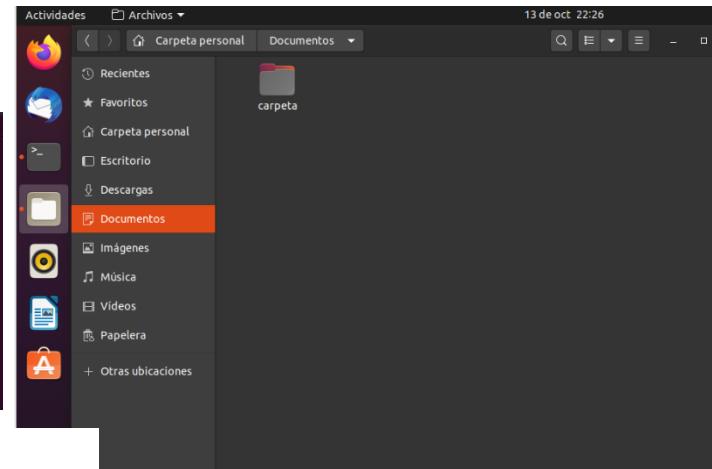


Ilustración 77. Ejemplo2 antes de ejecutar el comando-carpeta

Capturas después de ejecutar el comando

```
victor@victor-VirtualBox:~/Escritorio$ ls
[compartida] holamundo.c
victor@victor-VirtualBox:~/Escritorio$ mv holamundo.c /home/victor/Documentos
victor@victor-VirtualBox:~/Escritorio$
```

Ilustración 78. Ejemplo2 comando ejecutado-consola

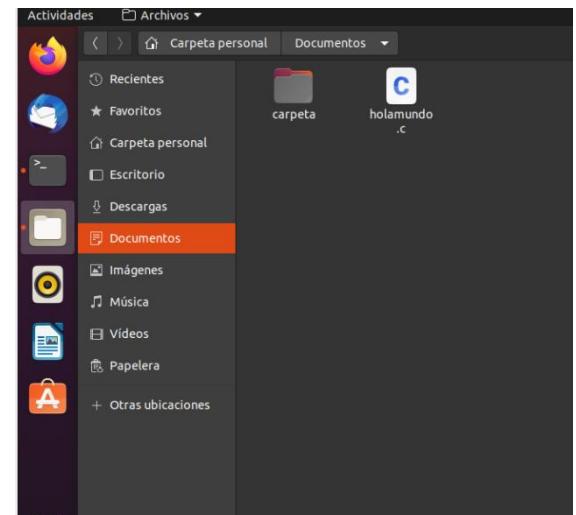


Ilustración 79. Ejemplo2 comando ejecutado-carpeta

### Ejemplo 3

En este caso realizaremos el movimiento de dos ficheros distintos a la vez, del fichero Documentos a Escritorio, para eso usamos el siguiente comando: mv carpeta holamundo.c /home/victor/Escritorio.

Capturas antes de ejecutar el comando.

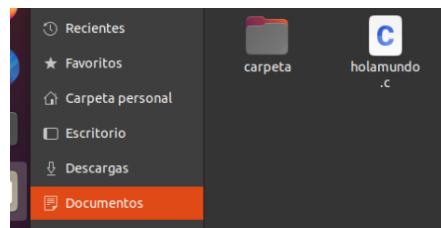


Ilustración 81. Antes de ejecutar el comando-carpeta

```
victor@victor-VirtualBox:~/Documentos$ ls
carpeta holamundo.c
victor@victor-VirtualBox:~/Documentos$
```

Ilustración 80. Ejemplo3 antes de ejecutar el comando-consola

Capturas después de ejecutar el comando.

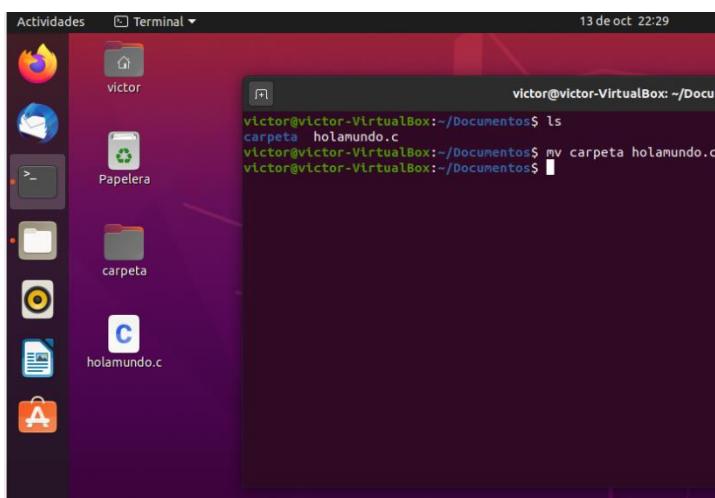


Ilustración 82. Ejemplo3 comando ejecutado-consola

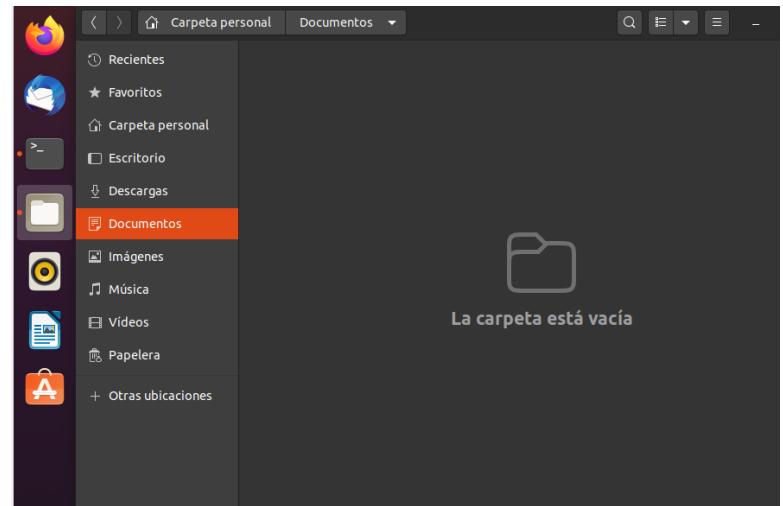
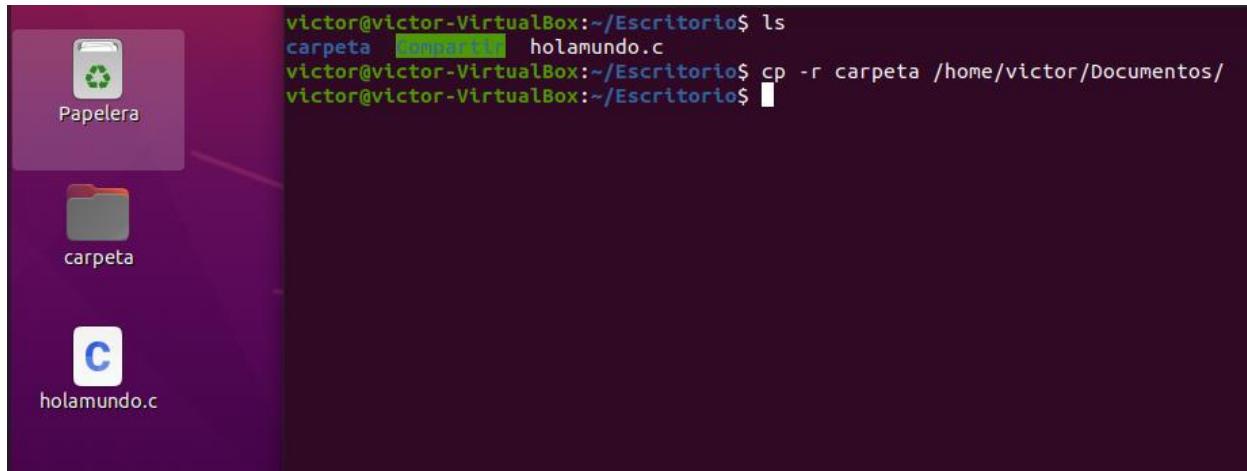


Ilustración 83. Ejemplo3 comando ejecutado-carpeta

#### Ejemplo 4.

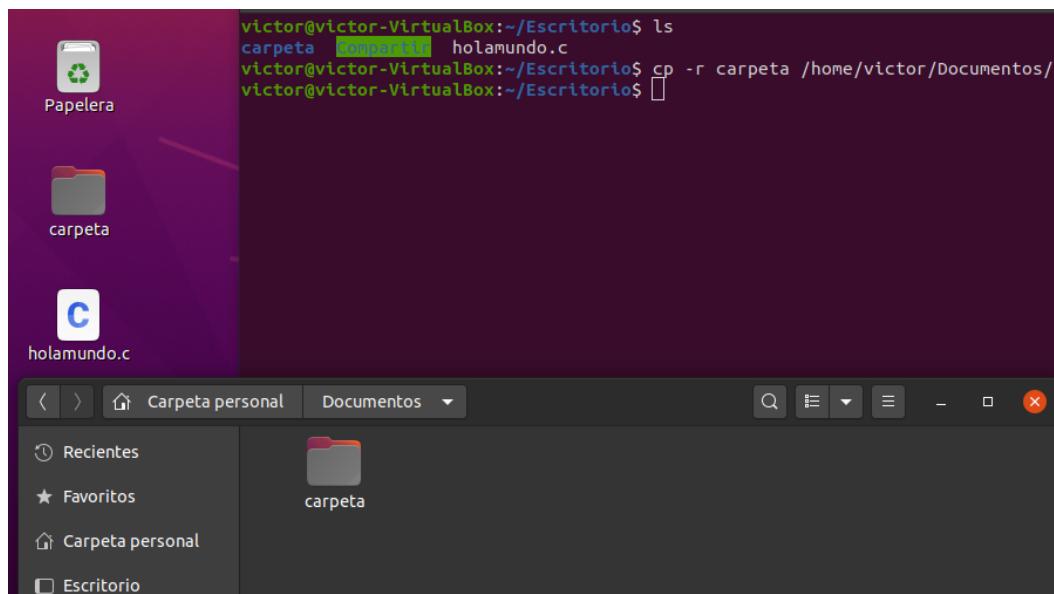
Ahora realizaremos una copia de los documentos antes utilizados, previamente fueron eliminados del directorio Documentos y únicamente se encuentran en el Escritorio, ahora procederemos a realizar una copia del archivo al fichero Documentos, el comando que usaremos será cp -r carpeta /home/victor/Documentos. Captura antes de ejecutar el comando.



```
victor@victor-VirtualBox:~/Escritorio$ ls
carpeta Compartida holamundo.c
victor@victor-VirtualBox:~/Escritorio$ cp -r carpeta /home/victor/Documentos/
victor@victor-VirtualBox:~/Escritorio$
```

Ilustración 84. Ejemplo4 antes de ejecutar el comando

Captura una vez ejecutado el comando anterior.



```
victor@victor-VirtualBox:~/Escritorio$ ls
carpeta Compartida holamundo.c
victor@victor-VirtualBox:~/Escritorio$ cp -r carpeta /home/victor/Documentos/
victor@victor-VirtualBox:~/Escritorio$
```

Ilustración 85. Ejemplo4 comando ejecutado

## Ejemplo5.

Este ejercicio realizara la copia de dos archivos distintos de manera simultánea, del fichero Documentos al de Escritorio. El comando que usaremos es: cp -r carpeta holamundo.c /home/victor/Escritorio/. Capturas antes y después ejecutar el comando.

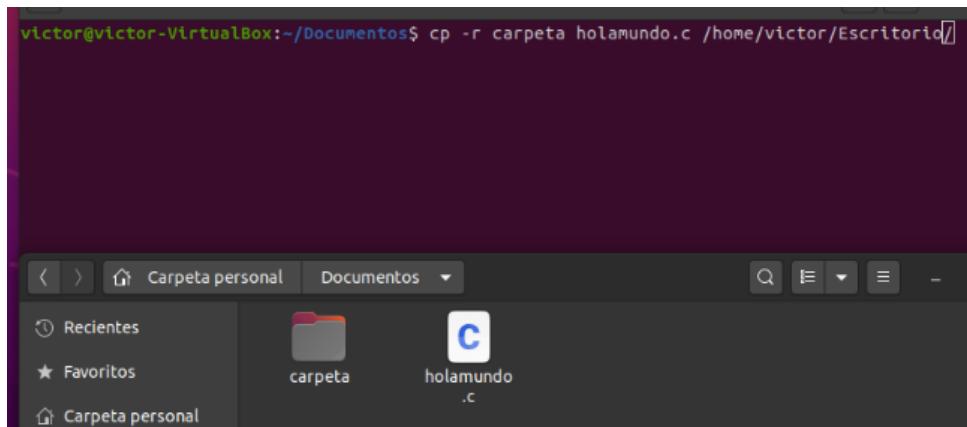


Ilustración 87. Ejemplo5 antes de ejecutar el comando

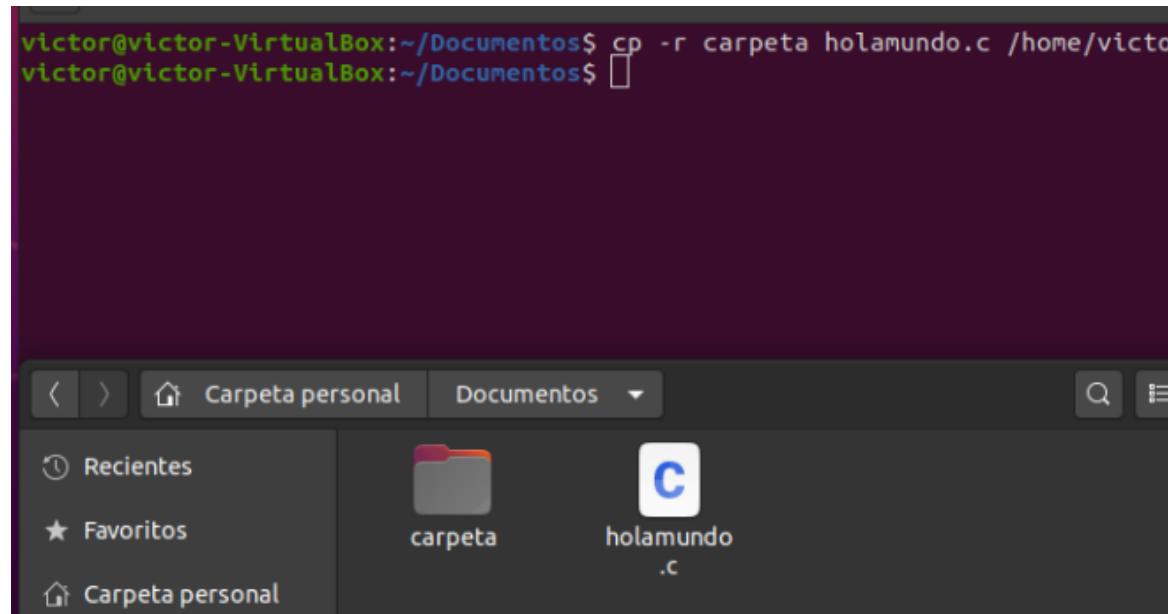


Ilustración 86. Ejemplo5 comando ejecutado

## Secuencia de pasos para crear, verificar y eliminar un usuario

Existen dos diferentes formas de crear un nuevo usuario desde la terminal de Linux. Procedemos a mostrar la primera forma de crear un nuevo usuario.

### Método 1 para crear un nuevo usuario

Para comenzar debemos realizar estos comandos como usuario root, para ello tecleamos el siguiente comando en la terminal: sudo -s, nos solicitará la contraseña del super usuario y después de ello podremos continuar. Una forma de identificar que estamos operando como super usuario es que en la terminal el último símbolo que se muestra es #.

```
victor@victor-VirtualBox:~$ sudo -s
[sudo] contraseña para victor:
root@victor-VirtualBox:/home/victor#
```

Ilustración 88. Super usuario

Ya en modo root procedemos a teclear el siguiente comando: adduser <nombre\_usuario>, para nuestro caso el comando quedó de la siguiente manera: adduser prueba, donde prueba será el nombre de nuestro nuevo usuario.

```
root@victor-VirtualBox:/home/victor# adduser prueba
Añadiendo el usuario `prueba' ...
Añadiendo el nuevo grupo `prueba' (1001) ...
Añadiendo el nuevo usuario `prueba' (1001) con grupo `prueba' ...
Creando el directorio personal `/home/prueba' ...
Copiando los ficheros desde `/etc/skel' ...
Nueva contraseña: [REDACTED]
```

Ilustración 89. Agregando un nuevo usuario

Se mostrarán los campos que nos solicitan datos del nuevo usuario, pero el único campo obligatorio es la contraseña para el usuario, en esta sección podemos omitir estos campos con solo dar enter, o si lo deseamos podemos proporcionar los datos requeridos.

Al momento de terminar de llenar los campos preguntara si los datos proporcionados son correctos, de ser el caso presionamos S para terminar la creación del nuevo usuario.

```
root@victor-VirtualBox:/home/victor# adduser prueba
Añadiendo el usuario `prueba' ...
Añadiendo el nuevo grupo `prueba' (1001) ...
Añadiendo el nuevo usuario `prueba' (1001) con grupo `prueba' ...
Creando el directorio personal `/home/prueba' ...
Copiando los ficheros desde `/etc/skel' ...
Nueva contraseña:
Vuelva a escribir la nueva contraseña:
passwd: contraseña actualizada correctamente
Cambiando la información de usuario para prueba
Introduzca el nuevo valor, o presione INTRO para el predeterminado
    Nombre completo []: Norberto
    Número de habitación []: 2
    Teléfono del trabajo []: 5543434343
    Teléfono de casa []: 12435678
    Otro []:
¿Es correcta la información? [S/n] s
root@victor-VirtualBox:/home/victor# exit
exit
victor@victor-VirtualBox:~$
```

Ilustración 90. Agregando un nuevo usuario, confirmación

## Método 2 para crear un nuevo usuario.

Este método evita el formulario que vimos en el método anterior y la principal ventaja de ello es que cuando se requiere la creación de diferentes usuarios de forma rápida, este método permite ahorrar mucho tiempo.

De igual forma que el método anterior, todos los comandos que introducimos fueron realizamos desde el usuario root, el comando utilizado es: useradd, dicho comando requiere ciertos parámetros y uno de ellos es el equipo de trabajo, este debe existir antes de realizar la creación del nuevo usuario. Para crea un nuevo grupo ejecutamos el comando: *groupadd <nombre\_grupo>*. En nuestro caso Miequipo es el nombre del nuevo equipo.

```
root@victor-VirtualBox:/home/victor# groupadd Miequipo
```

Ilustración 91. Creando un nuevo grupo

Después de crear el nuevo equipo, procedemos a crear el nuevo usuario de la siguiente manera:  
*useradd -g Miequipo -m David* donde: -g significa que al nuevo usuario le asignaremos un grupo.

Miequipo es el nombre del equipo al que asignamos el usuario, -m crea el directorio home del usuario, si este atributo no se especifica, el usuario no podrá iniciar sesión.

David será el nombre del nuevo Usuario.

Si después de dar enter solo se muestra una nueva línea, eso significa que el nuevo usuario se creó de forma exitosa. Para corroborarlo verificaremos que el nuevo usuario haya sido creado de la siguiente forma.

```
root@victor-VirtualBox:/home/victor# groupadd Miequipo  
root@victor-VirtualBox:/home/victor# useradd -g Miequipo -m David  
root@victor-VirtualBox:/home/victor#
```

Ilustración 92. Creando un nuevo usuario a partir del grupo

## Verificación de la existencia de usuarios

De igual forma existen distintas formas de corroborar que los usuarios antes creados, realmente están en el sistema.

### Método 1 forma gráfica.

Esta es la forma más sencilla pues hacemos uso de la interfaz gráfica de usuario de Ubuntu, lo único que necesitamos hacer es cerrar sesión e iniciar desde el nuevo usuario que hemos creado. Si el usuario se creó exitosamente, entonces debemos ver los usuarios en el inicio de sesión.

### Método 2 Terminal



Ilustración 93. Verificar usuarios forma gráfica

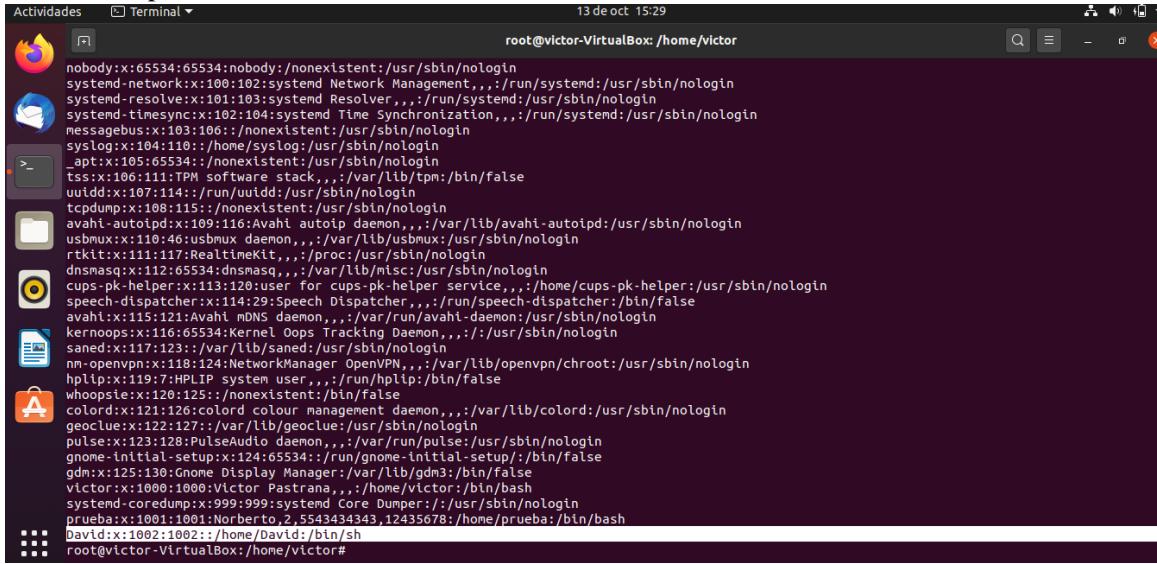
Otra de las formas de verificar que los usuarios que hemos creado se hayan añadido al sistema de manera correcta es mediante el uso de la terminal con la ejecución del siguiente comando:

```
cat /etc/passwd
```

```
victor@victor-VirtualBox:~$ sudo -s  
[sudo] contraseña para victor:  
root@victor-VirtualBox:/home/victor# cat /etc/passwd
```

Ilustración 94. Verificar usuarios desde terminal

Y se mostrara la siguiente pantalla donde se aprecian los usuarios del sistema. Al final de lo que muestra se aprecian ambos usuarios creados anteriormente.



```
nobody:x:65534:65534:nobody:/usr/sbin/nologin
systemd-network:x:100:102:system Network Management,,,:/run/systemd:/usr/sbin/nologin
systemd-resolve,x:101:103:system Resolver,,,:/run/systemd:/usr/sbin/nologin
systemd-timesync,x:102:104:system Time Synchronization,,,:/run/systemd:/usr/sbin/nologin
messagebus,x:103:106:/nonexistent:/usr/sbin/nologin
syslog,x:104:110:/home/syslog:/usr/sbin/nologin
_apt,x:105:65534:/nonexistent:/usr/sbin/nologin
tss,x:106:111:TPM software stack,,,:/var/lib/tpm:/bin/false
uuid,x:107:114:/run/uuid:/usr/sbin/nologin
tcpdump,x:108:115:/nonexistent:/usr/sbin/nologin
avahi-autoipd,x:109:116:Avahi AutoIP Daemon,,,:/var/lib/avahi-autoipd:/usr/sbin/nologin
usbmux,x:110:46:usbmux daemon,,,:/var/lib/usbmux:/usr/sbin/nologin
rtkit,x:111:117:RealtimeKit,,,:/proc:/usr/sbin/nologin
dnsmasq,x:112:65534:dnsmasq,,,:/var/lib/misc:/usr/sbin/nologin
cups-pk-helper,x:113:120:user for cups-pk-helper service,,,:/home/cups-pk-helper:/usr/sbin/nologin
speech-dispatcher,x:114:29:speech Dispatcher,,,:/run/speech-dispatcher:/bin/false
avahi,x:115:121:Avahi mDNS daemon,,,:/var/run/avahi-daemon:/usr/sbin/nologin
kernoops,x:116:65534:KernelOops Tracking Daemon,,,:/usr/sbin/nologin
saned,x:117:123:/var/lib/saned:/usr/sbin/nologin
nm-openvpn,x:118:124:NetworkManager OpenVPN,,,:/var/lib/openvpn/chroot:/usr/sbin/nologin
hpiloip,x:119:7:HPLiP system user,,,:/run/hpiloip:/bin/false
whoopsie,x:120:125:/nonexistent:/bin/false
colorld,x:121:126:color colour management daemon,,,:/var/lib/colorld:/usr/sbin/nologin
geoclue,x:122:127:/var/lib/geoclue:/usr/sbin/nologin
pulse,x:123:128:PulseAudio daemon,,,:/var/run/pulse:/usr/sbin/nologin
gnome-initial-setup,x:124:65534:/run/gnome-initial-setup:/bin/false
gdm,x:125:130:Gnome Display Manager:/var/lib/gdm3:/bin/false
victor,x:1000:1000:Victor Pastrana,,,:/home/victor:/bin/bash
systemd-coredump,x:999:999:systemd Core Dumper:/:/usr/sbin/nologin
prueba,x:1001:1001:Norberto,2,5543434343,12435678:/home/prueba:/bin/bash
David,x:1002:1002::/home/David:/bin/sh
root@victor-VirtualBox:/home/victor#
```

Ilustración 95. Ejecución de la verificación

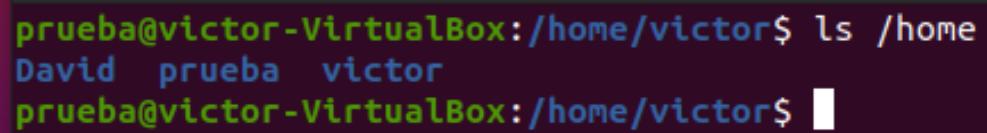
En la parte inferior se ven los usuarios con sus respectivos atributos.

```
gdm:x:125:130:Gnome Display Manager:/var/lib/gdm3:/bin/false
victor:x:1000:1000:Victor Pastrana,,,:/home/victor:/bin/bash
systemd-coredump,x:999:999:systemd Core Dumper:/:/usr/sbin/nologin
prueba,x:1001:1001:Norberto,2,5543434343,12435678:/home/prueba:/bin/bash
David,x:1002:1002::/home/David:/bin/sh
root@victor-VirtualBox:/home/victor#
```

Ilustración 96. Usuarios con sus atributos

Método 3 Terminal.

Otra forma para verificar cuantos usuarios existe en el sistema, es realizar un listado de los usuarios en el fichero home. Solo necesitamos ejecutar el siguiente comando: ls.



```
prueba@victor-VirtualBox:/home/victor$ ls /home
David prueba victor
prueba@victor-VirtualBox:/home/victor$
```

Ilustración 97. Verificación de usuarios con ls

## Cambiar de usuario.

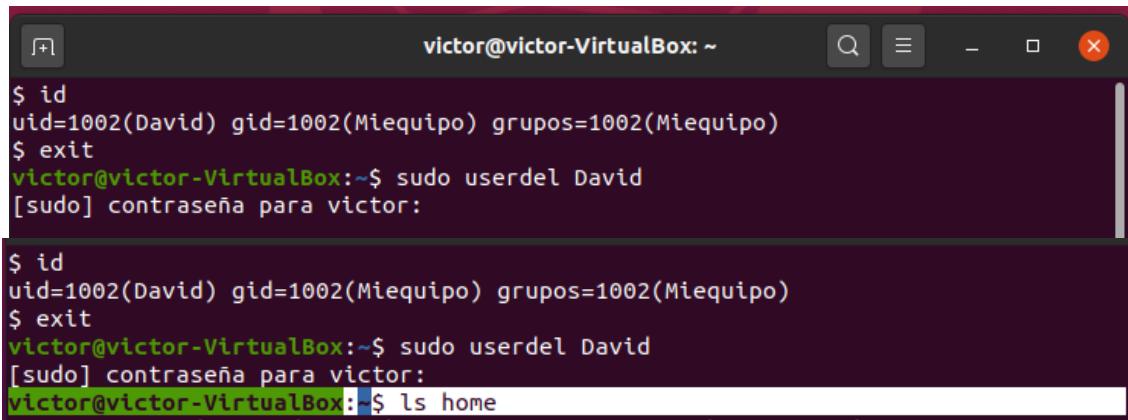
Para realizar un cambio de sesión necesitamos conocer la contraseña del usuario a cuya cuenta deseamos entrar. Conociendo ese dato lo único que requerimos hacer es ejecutar el siguiente comando: su <nombre\_usuario>. En este caso se realizó de la siguiente forma donde observamos que incluso el título de la terminal cambia de usuario.

## Eliminar un usuario.

```
victor@victor-VirtualBox:~$ su prueba  
Contraseña:  
prueba@victor-VirtualBox:/home/victor$ sudo userdel -r prueba
```

Ilustración 98. Cambiar de usuario

Para suprimir una cuenta de usuario, necesitamos contar con los privilegios del súper usuario y utilizar el siguiente comando userdel <nombre\_usuario>. Para este ejercicio eliminamos al usuario David. Una vez que escribimos el comando necesitamos ingresar la contraseña del súper usuario y el usuario quedará eliminado



The screenshot shows a terminal window titled 'victor@victor-VirtualBox: ~'. The user first runs 'id' to check their current user information, which shows they are 'uid=1002(David)'. They then run 'exit' to log out. After logging back in, they run 'sudo userdel David'. The terminal prompts for the password with '[sudo] contraseña para victor:'. Finally, they run 'ls home' to verify that the user 'David' has been removed from the system.

```
$ id  
uid=1002(David) gid=1002(Miequipo) grupos=1002(Miequipo)  
$ exit  
victor@victor-VirtualBox:~$ sudo userdel David  
[sudo] contraseña para victor:  
  
$ id  
uid=1002(David) gid=1002(Miequipo) grupos=1002(Miequipo)  
$ exit  
victor@victor-VirtualBox:~$ sudo userdel David  
[sudo] contraseña para victor:  
victor@victor-VirtualBox:~$ ls home
```

Ilustración 99. Eliminar un usuario

Finalmente para verificar que el usuario fue eliminado podemos revisar los usuarios que estén el sistema con alguno de los métodos que vimos hace un momento, nosotros ocupamos el segundo método. En esta parte el usuario David ya no aparece más en la lista de usuarios.

```
hplip:x:119:7:HPLIP system user,,,:/run/hplip:/bin/false
whoopsie:x:120:125::/nonexistent:/bin/false
colord:x:121:126:colord colour management daemon,,,:/var/lib/colord:/usr/sbin/no
login
geoclue:x:122:127::/var/lib/geoclue:/usr/sbin/nologin
pulse:x:123:128:PulseAudio daemon,,,:/var/run/pulse:/usr/sbin/nologin
gnome-initial-setup:x:124:65534::/run/gnome-initial-setup/:/bin/false
gdm:x:125:130:Gnome Display Manager:/var/lib/gdm3:/bin/false
victor:x:1000:1000:Victor Pastrana,,,:/home/victor:/bin/bash
systemd-coredump:x:999:999:systemd Core Dumper:/:/usr/sbin/nologin

```

Ilustración 100. Verificación de eliminar un usuario

## Compilar Hola Mundo en código para nano.

Creación Hola Mundo

```
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ cd Documentos
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ nano hello.c
```

Ilustración 101. Creación de un Hola Mundo

Código Hola Mundo editado en nano

```
GNU nano 4.8                                     hello.c
#include <stdio.h>

int main(){
    printf("Hola mundo \n");
}
```

Ilustración 102. Código Hola mundo editado en Nano

Código Hola Mundo compilación nano

```
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ nano hello.c
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ nano hello.c
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ nano hello.c
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ gcc hello.c -o hello1
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ ./hello1
Hola mundo
```

Ilustración 103. Compilación nano del código Hola Mundo

## Compilar Hola Mundo en código para vi.

Creación Hola Mundo

```
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ cd Documentos  
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ vi hello2.c
```

Ilustración 104. Creación Hola mundo para vi

Edición Hola Mundo en vi

```
#include <stdio.h>  
  
int main(){  
  
    printf("Hola Mundo \n");  
    return 0;  
}
```

Ilustración 105. Edición Hola Mundo en vi

Ejecución hola mundo en vi

```
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ vi hello2.c  
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ gcc hello2.c -o hola2  
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ ./hola2  
Hola Mundo  
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$
```

Ilustración 106. Ejecución Hola Mundo en vi

## Compilar Hola Mundo en código para pico.

Instalación de pico (Pine Composer).

Utilizando el gestor de los paquetes para la instalación de PICO podemos darnos cuenta de que el paquete no puede ser localizado por el gestor.

```
ubuntu@ubuntu:~$ sudo apt install alpine-pico  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
E: Unable to locate package alpine-pico
```

Ilustración 107. Instalación de pico

Revisión de los comandos de instalación.

Debemos actualizar nuestro apt para poder instalarlo, todo con base a esta referencia [15].

Install alpine-pico by entering the following commands in the terminal:

```
sudo apt update  
sudo apt install alpine-pico
```

Description:

*Simple text editor from Alpine, a text-based email client*

"pico" is a simple but powerful text editor. It was originally the pine command in context and is now used as a stand-alone editor by many users. . It is s

Homepage: <http://alpine.freeiz.com/alpine/>

*Ilustración 108. Comandos de instalación pico*

## Actualización de apt.

Ya he actualizado el apt, pero haciendo la prueba de nuevo el error continuo para la instalación de nuestro editor pico.

```
ubuntu@ubuntu:~$ sudo apt-get update  
Ign:1 cdrom://Ubuntu 20.04.1 LTS _Focal Fossa_ - Release amd64 (20200731) focal  
InRelease  
Hit:2 cdrom://Ubuntu 20.04.1 LTS _Focal Fossa_ - Release amd64 (20200731) focal  
Release  
Hit:4 http://archive.ubuntu.com/ubuntu focal InRelease  
Get:5 http://security.ubuntu.com/ubuntu focal-security InRelease [107 kB]  
Get:6 http://archive.ubuntu.com/ubuntu focal-updates InRelease [111 kB]  
Get:7 http://security.ubuntu.com/ubuntu focal-security/main amd64 DEP-11 Metadata [24.3 kB]  
Get:8 http://archive.ubuntu.com/ubuntu focal-updates/main amd64 Packages [610 kB]  
]  
Get:9 http://archive.ubuntu.com/ubuntu focal-updates/main amd64 DEP-11 Metadata [229 kB]  
Get:10 http://archive.ubuntu.com/ubuntu focal-updates/restricted amd64 Packages [75.9 kB]  
Fetched 1,158 kB in 4s (312 kB/s)  
Reading package lists... Done
```

*Ilustración 109. Actualización del apt*

## Buscando soporte para pico en sitios oficiales.

Revisando el sitio oficial, nos damos cuenta de que el sitio oficial esta caído.



*Ilustración 110. Soporte para pico*

En el siguiente sitio anglosajón [16] encontramos más información sobre pico, algunos enlaces oficiales y los archivos binarios para la instalación manual, pero absolutamente todos los links estaban fuera de servicio. Los archivos binarios nos hubieran permitido instalar el editor sin necesidad de nuestro gestor de paquetes.

## Pico Distribution/Download

- a) pine requires the pico libraries,
- b) there is no separate pico distribution, and
- c) the default make files build both pine and (stand alone) pico.

Summary: You have to download the pine distribution file to install pico. Yes, the distribution of pine-4.10 compressed with GNU zip is about 2.7Mb. I think that's too much to download for installing a simple editor. ;-) But that's why there are precompiled versions of pico (and pine) on the UW server with compressed binaries.

PICO Distribution -> PINE Home Page  
<http://www.washington.edu/pine/>  
 PICO is distributed with the newsreader [PINE](#). The following page tells where to get sources and binaries of PINE (which include PICO):  
<http://www.washington.edu/pine/overview/availability.html>  
 Binaries for DOS:  
<ftp://ftp.uni-koeln.de/pc/msdos/editors/dosjoe10.zip> (DJGPP)  
<ftp://ftp.simtel.net/pub/simtelnet-gnu/djgpp/v2apps/> [1999-07-26]

```
pico396b.zip 228 Kb Thu Sep 25 00:00:00 1997
pico396s.zip 185 Kb Thu Jun 25 00:00:00 1998
```

Binaries for Unix: [1997-01-06,1998-05-14]  
<ftp://ftp.cac.washington.edu/pine/unix-bin/>  
<ftp://ftp.cac.washington.edu/pine/unix-bin-compressed/>

*Ilustración 111. Pico Distribution/Download*

La alternativa “clon” de pico es nano, la diferencia es que nano es de código abierto y ya está incluido por defecto en distribuciones como lo es el caso de Ubuntu.

## Compilar Hola Mundo en código para GNU Emacs. (Para compensar pico)

Instalación de Emacs.

Ya hemos instalado Emacs por medio del gestor de paquetes.

```
ubuntu@ubuntu:~$ sudo add-apt-repository ppa:kelleky/emacs
This repository contains updated `emacs` packages based on stable

The following package series are available:
- `emacs27`: based on 27.x-series releases.
- `emacs26`: based on 26.x-series releases.
- `emacs25`: based on 25.x-series releases.
```

Ilustración 112. Instalación de emacs

```
ubuntu@ubuntu:~$ sudo apt install emacs26
Reading package lists... Done
Building dependency tree
Reading state information... Done
```

Ilustración 113. Instalación de emacs comando

Escritura del programa “Hola mundo”.

```
#include <stdio.h>

int main(){

    printf("Hola mundo \n");
    return 0;

}
```

Ilustración 114. Código del programa Hola mundo

Compilación y ejecución del programa dentro del editor.

Todo lo hacemos desde la consola de GNU Emacs.

```
ubuntu@ubuntu:~/Desktop$ gcc -o hola Holamundo.c
```

```
ubuntu@ubuntu:~/Desktop$ ./hola
Hola mundo /nubuntu@ubuntu:~/Desktop$ █
```

Ilustración 115. Ejecución del programa dentro de Emacs

## Código de programa1a.c

```
#include <stdio.h>
#include <stdlib.h>
int main(){
    printf("HOME:%s\n",getenv("HOME"));
    return 0;
}
```

## Resultado compilación Programa1a.c

```
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ nano Programa1a.c
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ gcc Programa1a.c -o Pr
ograma1a
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ ./Programa1a
HOME:/home/ricardomachorro
```

Ilustración 116. Compilación Programa1a.c

## Compilación y pantalla de programa1a.c

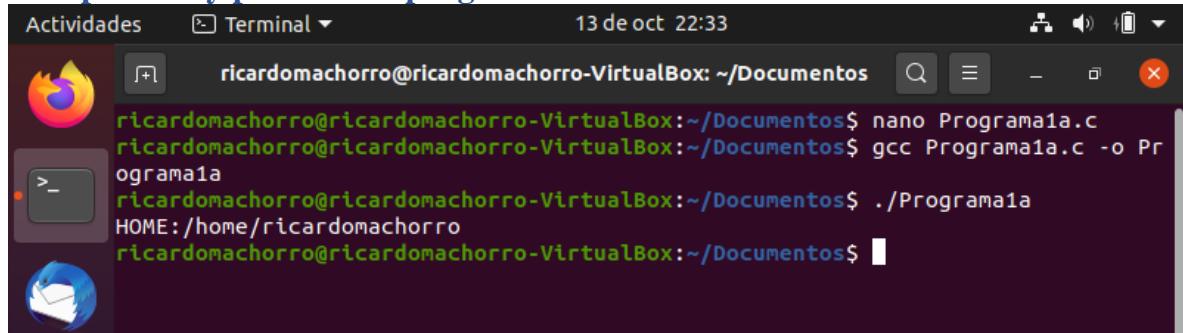


Ilustración 117. Compilación y pantalla de programa1a.c

## Código de programa2a.c

```
#include <stdio.h>
#include <stdlib.h>
int main(){
```

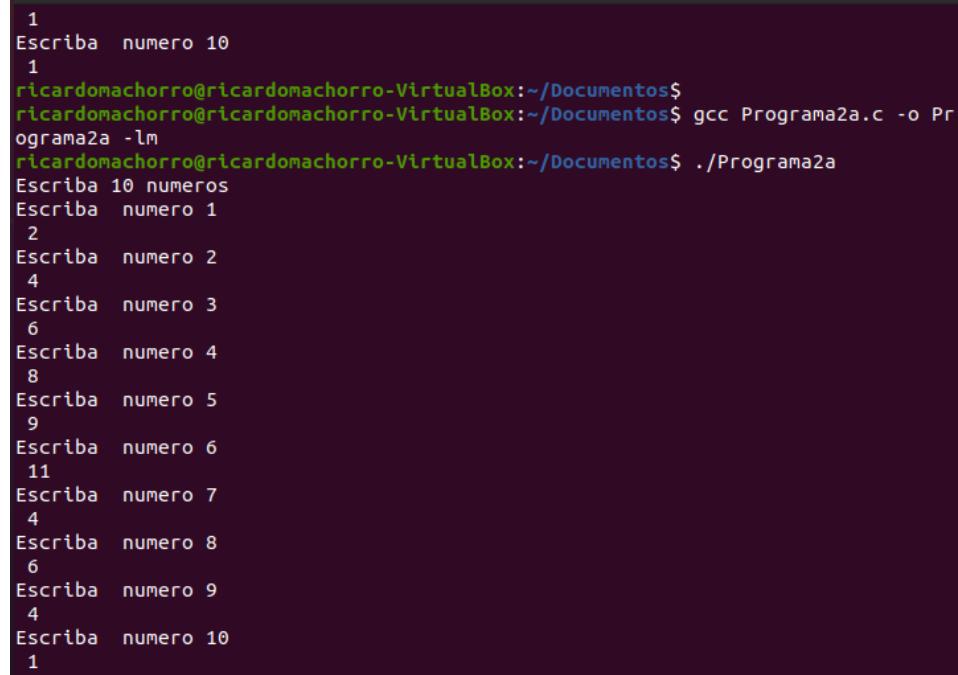
```

printf("HOME:%s\n",getenv("HOME"));

return 0;
}

```

## Compilación y pantalla de programa2a.c



```

1
Escriba numero 10
1
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ 
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ gcc Programa2a.c -o Programa2a -lm
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ ./Programa2a
Escriba 10 numeros
Escriba numero 1
2
Escriba numero 2
4
Escriba numero 3
6
Escriba numero 4
8
Escriba numero 5
9
Escriba numero 6
11
Escriba numero 7
4
Escriba numero 8
6
Escriba numero 9
4
Escriba numero 10
1

```

Ilustración 118. Compilación y pantalla de programa2a

1	Numero 1 : 2.000000
2	Numero 2 : 4.000000
3	Numero 3 : 6.000000
4	Numero 4 : 8.000000
5	Numero 5 : 9.000000
6	Numero 6 : 11.000000
7	Numero 7 : 4.000000
8	Numero 8 : 6.000000
9	Numero 9 : 4.000000
10	Numero 10 : 1.000000

Ilustración 119. Pantalla de programa2a

## Código de programa3a.c

```

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

```

```

#include <math.h>

int main(){

FILE *fp;
float suma=0;
float mult=1;
float var=0;
float des=0;
float prom=0;
float colNum[10];
char linea[50];
int cont1=0;
char *pnt;
char numeroStr[20];
fp=fopen("TextoPrograma2a.txt","r");
while(fgets(linea, sizeof(linea),fp)){
    if(cont1<10){
        pnt=strchr(linea,':');
        pnt++;
        colNum[cont1]=strtof(pnt,NULL);
        cont1++;
    }
}

for(int i=0; i<10;i++){
    suma=suma+colNum[i];
    mult=mult*colNum[i];
}

prom=suma/10;
for (int i=0; i<10;i++){
    var=var + pow(colNum[i],2);
}
var=var/10;

```

```

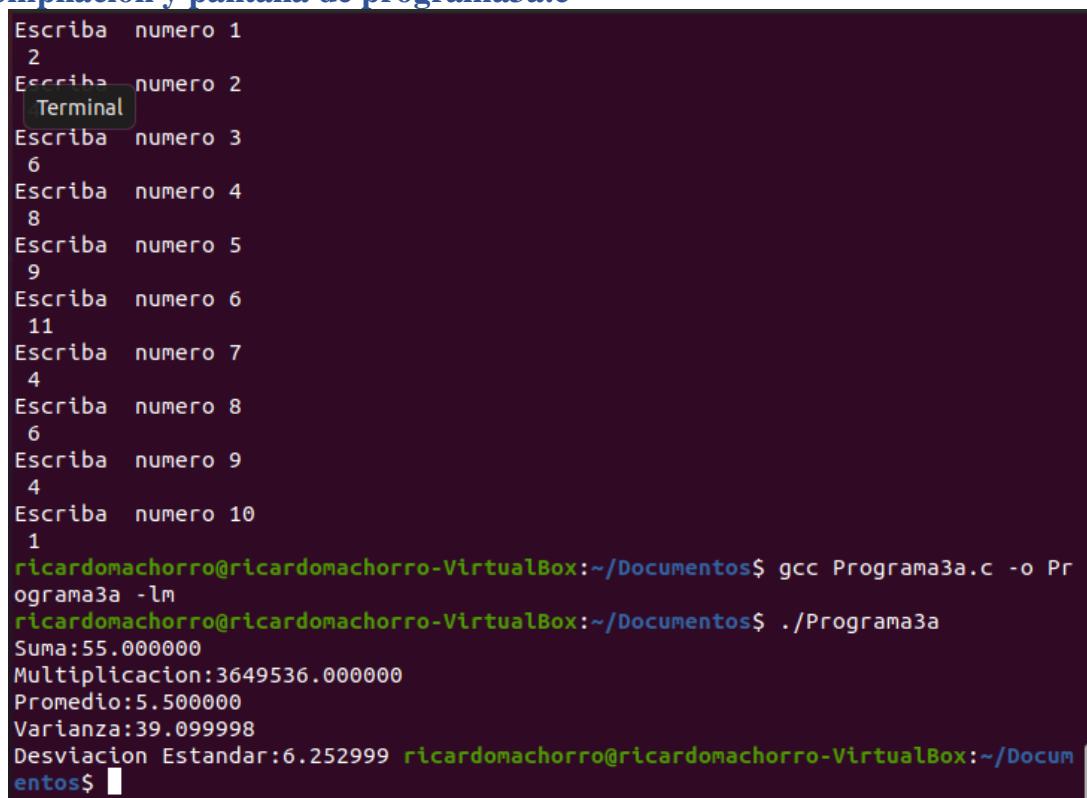
des=sqrt(var);

printf("Suma:%f \n",suma);
printf("Multiplicacion:%f \n",mult);
printf("Promedio:%f \n",prom);
printf("Varianza:%f \n",var);
printf("Desviacion Estandar:%f ",des);

return 0;
}

```

### Compilación y pantalla de programa3a.c



```

Escriba numero 1
2
Escriba numero 2
Terminal
Escriba numero 3
6
Escriba numero 4
8
Escriba numero 5
9
Escriba numero 6
11
Escriba numero 7
4
Escriba numero 8
6
Escriba numero 9
4
Escriba numero 10
1
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ gcc Programa3a.c -o Programa3a -lm
ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ ./Programa3a
Suma:55.000000
Multiplicacion:3649536.000000
Promedio:5.500000
Varianza:39.099998
Desviacion Estandar:6.252999 ricardomachorro@ricardomachorro-VirtualBox:~/Documentos$ █

```

Ilustración 120. Compilación y pantalla de programa3a

## **Conclusiones.**

### **Chavarría Vázquez Luis Enrique.**

En primera instancia, aprendí bastantes comandos de los cuales algunos ya eran viejos conocidos, pero por otra parte también había bastantes comandos de los cuales yo no tenía siquiera idea que existieran dentro de Linux, esto considero que es de vital importancia porque es bastante útil en la práctica tener cierto repertorio de ellos con el fin de ahorrar tiempo y volver la experiencia en las distribuciones Linux mucho más amena y sobre todo eficiente.

Habiendo dicho esto, quiero aclarar que con todo lo que estuvimos haciendo no me cabe duda del porque Linux es usado en entornos profesionales, pienso que con todas las facilidades que da para el desarrollo, acceso de datos y más a parte la enorme cantidad de comandos dedicados es sin temor a equivocarme muy eficaz y sobre todo sencillo después de un tiempo adaptándose.

Me agrado que después de un tiempo de haber entrado a la zona de confort en Microsoft tengo la oportunidad de volver a explotar la distribución de Ubuntu, aunque en la parte pico tuvimos problemas para instalar el editor considero que poder hacer la mayoría de cosas desde el gestor de paquetes es una maravilla, cuestión que en Windows no es para nada común, de hecho, lo más común en Windows es directamente emplear los instaladores guiados, pero en Linux es muy cómodo. En lo que a pico respecta, considero que es muy triste que la página oficial y los links de descarga de los archivos binarios estuvieran rotos así como lo estaba el paquete en el gestor, pero afortunadamente tenemos nano que en esencia tiene las mismas prestaciones y además es de código cien por ciento libre lo cual al final del día representa una ventaja tremenda.

No esta demás decir que ya llevaba algo de tiempo sin trabajar en C, sé que hasta cierto punto C ofrece muchas ventajas en rendimiento y me siento feliz de volver a usar este lenguaje, ya que había estado muy metido en JavaScript, Python y sobre todo en mis tiempos libres trabajo mucho con Rust y una variante de JS llamada Typescript, pero C fue el primer lenguaje en el que aprendí a programar por lo que es como volver a casa después de un largo tiempo, con esto quiero decir que me siento congratulado de poder volver a desarrollar proyectos en este lenguaje.

Sin duda de los tres editores que empleamos en la práctica mi favorito personal es Vim por las facilidades que ofrece, dentro de lo cual, hablando de facilidades me sorprendió mucho saber que las variables globales provienen de precisamente Linux, de hecho, yo creí que era algo original de

Microsoft, pero a decir verdad me fascina como la comunidad de Linux sigue aportando innovaciones que posteriormente llegan al terreno privado lo cual está muy bien.

La seguridad en Linux es muy buena, cosa que podemos ver cuándo vamos a emplear algún tipo de recurso adicional nos pide ciertos permisos, esto lo estuvimos discutiendo como equipo y para ser honestos considero que es bueno, no por nada Linux es uno de los entornos más seguros; aunque puede ser un poco tedioso, por la forma en que cuido mi sistema pienso que a la larga nos puede evitar conflictos de seguridad y nos ayuda a pensar bien que herramientas estamos usando, esto nos ocurrió con math en C cuando estuvimos desarrollando los ejercicios.

Algo de lo que jamás me cansare es la comunidad de Linux y la tremenda cantidad de información que esta misma sube a la red lo menciono porque durante la práctica tuvimos que ayudar a un compañero por problemas con su configuración para poder realizar ciertas operaciones al final gracias a la guía de la comunidad GNU eso lo saco del aprieto y él nos platicó su experiencia; con esto se aprecia como la comunidad del entorno es por si sola muy valiosa y es un punto que debería motivar a más a usar Linux.

Finalmente quiero cerrar aclarando que me gusta mucho que podamos trabajar con consola directamente, hace tiempo yo use Arch y en esta distribución uno puede elegir si instalar in gestor de ventanas para las consolas o directamente trabajar a punta de líneas de código en la consola, aunque al comienzo es difícil acostumbrarse al final se vuelve algo muy bueno en términos de velocidad, quiero cerrar diciendo que lo bueno de trabajar principalmente con la terminal es la velocidad pero sobre todo la satisfacción de conocer dichos “atajos” y estar sacando el máximo partido al sistema.

## **Juárez Espinoza Ulises.**

Como usuario de Windows yo soy una de las personas a las que el manejo de la computadora se le hace fácil, si quiero realizar una acción como abrir una aplicación, basta con ir al buscador escribir alguna referencia y darle clic a la aplicación deseada. Nunca me había preguntado porque esta acción es tan sencilla y mucho menos si era igual en otro sistema operativo, al decir verdad siempre me he mantenido en mi burbuja de Windows, no es que los demás sistemas operativos se me hagan buenos o malos, simplemente por la facilidad y comodidad que me ofrece Windows me ha sido suficiente para nunca haber considerado cambiar de sistema operativo. Los pocos acercamientos que había tenido con otros sistemas como por ejemplo Linux, han sido meramente por cuestiones académicas. Esta práctica me ayudo a comprender que la facilidad con la que manejamos a un sistema operativo es la interfaz gráfica, mediante el uso de imágenes y objetos gráficos nos muestra las acciones disponibles en el sistema. Todo está muy bien si se es un usuario común, pero comprendo que siendo estudiante de ingeniería en sistemas él no conocer mi sistema operativo y como decirle con comandos las acciones que debe realizar no se puede justificar. Así como también soy consciente que el sistema que se usa para desarrollar en el campo laboral es principalmente Linux, me llevaría mucho tiempo explicar los motivos por los que es tan apto y usado para esta actividad, a grandes rasgos, es de código abierto, no tiene restricciones, es configurable y de lo que vamos mucha disposición de herramientas de programación.

Considero que la razón por la que no es tan usado por usuarios comunes es por la dificultad que puede representar comunicarse con el sistema operativo si no se dispone de una interfaz gráfica y no están dispuestos a aprender a usar el Shell, que a mi consideración es la herramienta más poderosa que tiene Linux para realizar las tareas que el usuario desea incluso por encima de su interfaz gráfica.

Como desarrollador constantemente estamos navegando entre carpetas, necesitamos compilar código, instalar herramientas, descargar librerías, etc. Esta práctica me ayudo a comprender como realizar mucho de esto, desde cómo acceder a carpetas con rutas relativas o absolutas, como redireccionar y los usos entre `>` y `>>`, hasta una gran variedad de comandos que siendo honestos ni sabía que existían, al decir verdad solo conocía los que había necesitado para realizar las actividades que me había solicitado en mi experiencia académica. Toda esta variedad de comandos me deja ver la libertad que ofrece este sistema operativo para manipularlo y porque es tan relevante. En lo que respecta a una de las partes que más nos interesan como programadores, que es editar código desde Linux, en un principio puede verse complicada pero sobre la marcha se fue haciendo sencillo, me llamo la atención el hecho que aunque no sea como en otros sistemas operativos, cuenta con editores nativos de Linux como vi y nano, y con muchos otros que se pueden instalar como es el caso de Emacs y vim, esta

parte me dejo bastante satisfecho, ya que como usuarios valoras mucho la variedad de opciones que tienes para realizar un acción, porque así eliges la que más te guste.

Esas han sido las conclusiones más importantes que me ha dejado esta práctica, estoy expectante a las siguientes y con ganas de seguir aprendiendo sobre este sistema operativo que hasta hace poco había sido enigmático para mí.

## **Machorro Vences Ricardo Alberto.**

Esta práctica me enseño que primera instancia que Linux tienen muchas funciones que le hacen merecedor de su reputación de la caja de herramientas de los sistemas operativos, permitiendo así poder expandir no solo el conocimiento técnico del funcionamiento y control real de los sistemas operativos sino que también permite ayudar a familiarizarse con el entorno de la consola, aunque este grado de libertad también puede ser algo intimidador para el usuario, como de hecho fue mi caso al principio ya que yo había estado acostumbrado al seguro y hermético Windows.

Además, la práctica me ayudo en cierto grado a acostumbrarme y aprender a usar más la consola y no usar atajos como podría ser copiar el texto o usar comandos anteriores para ahorrar tiempo, que creo que es muy importante porque si no se hace de manera manual o propia no se aprende verdaderamente lo que hace inútil la práctica hasta cierto grado cosa que afecta más al alumno que al maestro.

Algo que también aprendí de esta práctica es que Linux pide muchos permisos incluso cuando se ejecutan comandos que relativamente son sencillos, o con códigos en lenguaje C que no afectan en realidad al sistema o interactúan con un elemento importante de este, porque en uno de los códigos necesitaba usar la librería de `<math.h>` para usar las operaciones matemáticas de potencias o raíces cuadradas y la línea de comandos de Linux me preguntaba ser administrador, cosa que me extraño.

La práctica además me ayudo a practicar mis habilidades de programación en C porque yo no había codificado en este lenguaje en casi un año, ya que me había concentrado en la parte web de la programación con lenguajes como Dart o JavaScript y dentro de la parte más lógica me puse a aprender principalmente Python. Esto lo considero bueno porque muchos de los lenguajes mencionados anteriormente a diferencia de C tienen muchos métodos y estructuras de datos ya incluidas que facilitan acciones, pero que pueden hacer que se olvide como hacerlos.

En resumen, yo vi esta práctica como una forma no solo de poder expandir mi conocimiento en los temas que se solicitan en esta, sino que también ayudo a practicar habilidades y conocimientos de materias pasadas que por cuestiones de tiempo no he podido reforzar, y que además me ayudo a quitarme un poco el miedo que tengo a la consola y al sistema operativo y ver que en realidad este es una herramienta muy útil.

## **Pastrana Torres Victor Norberto.**

Gracias al desarrollo de esta práctica aprendí varias cosas acerca del funcionamiento de Linux, antes de esto mi único acercamiento con el sistema había sido en POO con la ejecución de algunos programas desde este sistema, pero con esa breve experiencia de uso puedo afirmar que el uso de Linux tiene ventajas si lo comparamos con Windows. Una de las ventajas que note al usar Linux es que los programas se desarrollan con mayor fluidez, por ejemplo en aquella unidad de aprendizaje desarrolle un programa que simulaba un sistema solar, y a momento de estarlo codificando en mi computadora con Windows note que el movimiento no era muy bueno, incluso llegaba a trabarse pero todos estos detalles desaparecieron cuando corrí el programa desde las computadoras de los laboratorios de ESCOM que cuentan con Ubuntu, el movimiento era muy bueno y no presentaba falla alguna.

Otro punto que encuentro a favor de Ubuntu es que al estar interactuando con el sistema mediante la terminal es más fácil que lo conozcas a profundidad, porque con tan solo hacer un cambio de fichero mediante referencia absoluta tienes que saber con exactitud hacia donde te diriges de lo contrario terminaras en otro lugar del sistema o también puede que el comando no funcione, caso contrario a un sistema como Windows es que en muchas ocasiones los usuarios no saben que cuentan con cierta carpeta.

Mientras llevaba a cabo la investigación documental de la práctica consulte varias fuentes que me ayudaron a comprender como funciona Linux y sobre todo perder el miedo a usar un nuevo sistema operativo desde la terminal. En primera instancia el usar una terminal podría parecer difícil y poco comprensible pero una vez que comienzas a interactuar y a “jugar” comprendes que tiene varias ventajas tales como la velocidad de respuesta, conocimiento de tu sistema, personalización y gestión de este.

Algunos de los problemas con lo que me enfrenté fue que al intentar crear un nuevo usuario decía que mi sistema tenía una configuración distinta y que yo como usuario no contaba con los derechos para hacer cambios de ese tipo. Fue entonces que busqué una guía para manejar Linux, y me apoye de la guía de GNU/Linux que me sacó de problemas y además vi a detalle el funcionamiento de algunos comandos.

Finalmente puedo concluir que con el desarrollo de esta práctica pude perder el miedo a usar la terminal, aprendí comandos algunos para interactuar con el sistema y navegar a través de él y además

recordé como codificar en lenguaje C ya que desde hace algunos meses no había desarrollado un programa en este lenguaje debido a que me concentré en otras tecnologías.

## Bibliografía

- [1] KDE, «KDE,» KDE , 15 Julio 2019. [En línea]. Available: [https://userbase.kde.org/What\\_is\\_KDE/es](https://userbase.kde.org/What_is_KDE/es). [Último acceso: 2020 Octubre 12].
- [2] GNOME, «GNOME,» GNOME, 2020. [En línea]. Available: <https://www.gnome.org/>. [Último acceso: 2020].
- [3] ORACLE, «ORACLE,» 2013. [En línea]. Available: <https://docs.oracle.com/cd/E19683-01/806-7612/startup-78447/index.html>. [Último acceso: 2020].
- [4] Wikipedia, «Wikipedia,» 2020. [En línea]. Available: [https://en.wikipedia.org/wiki/Command-line\\_interface](https://en.wikipedia.org/wiki/Command-line_interface). [Último acceso: 2020].
- [5] D. Ocean, «Digital Ocean,» 2020. [En línea]. Available: <https://www.digitalocean.com/community/tutorials/an-introduction-to-the-linux-terminal>. [Último acceso: 2020].
- [6] E. Seiver, «Linux in a nutshell,» de *Linux in a nutshell*, CDMX, O'Reilly, 2009.
- [7] ComputerNetworkingNotes, «ComputerNetworkingNotes,» ComputerNetworkingNotes, 2020. [En línea]. Available: <https://www.computernetworkingnotes.com/rhce-study-guide/types-of-users-in-linux-explained-with-accounts.html#:~:text=There%20are%20three%20types%20of,%2D%20root%2C%20regular%20and%20service..> [Último acceso: 2020].
- [8] SOSPEDIA, SOSPEDIA, 2020. [En línea]. Available: <https://sospedia.net/rutas-relativas-rutas-absolutas/>. [Último acceso: 2020].
- [9] DeNovatoANovato, «DeNovatoANovato,» DeNovatoANovato, 2020. [En línea]. Available: <https://denovatoanovato.net/rutas-relativas-y-rutas-absolutas/>. [Último acceso: 2020].
- [10] S. G. D., «Redireccionamiento en Linux,» 2020. [En línea]. Available:
  - [ ] <https://www.linuxtotal.com.mx/index.php?cont=redireccionamiento-en-linux>.
- [11] «Desde Linux.,» 2015. [En línea]. Available: <https://blog.desdelinux.net/mas-de-400-comandos-para-gnulinux-que-deberias-conocer/>.
- [12] D. A., «Variables de entorno de Linux: cómo leerlas y configurarlas en un VPS de Linux,» 25 Noviembre 2019. [En línea]. Available: <https://www.hostinger.mx/tutoriales/variables-de-entorno-linux-como-leerlas-y-configurarlas-vps/>. [Último acceso: 2020].
- [13] M. Ebrahim, «Aprenda Variables De Entorno De Linux Guía Paso A Paso,» 4 Febrero 2017. [En línea]. Available: <https://likegeeks.com/es/variables-de-entorno-de-linux/>. [Último acceso: 2020].
- [14] IONOS, «DIGITAL GUIDE IONOS,» 2018. [En línea]. Available:
  - [ ] <https://www.ionos.mx/digitalguide/servidores/configuracion/comandos-de-linux-la-lista-fundamental/>. [Último acceso: 2020].
- [15] How to install, «How to install,» 6 Octubre 2020. [En línea]. Available:
  - [ ] <https://www.howtoinstall.me/ubuntu/18-04/alpine-pico/>.
- [16] guckes, «guckes,» guckes, 9 Octubre 2020. [En línea]. Available:
  - [ ] <http://www.guckes.net/pico/>. [Último acceso: 2020].