Comandos básicos de Linux.

1. comando pwd

Usa el comando **pwd** para encontrar la ruta del directorio (carpeta) de trabajo actual en el que te encuentras. El comando devolverá una ruta absoluta (completa), que es básicamente una ruta de todos los directorios que comienzan con una barra diagonal (/) Un ejemplo de una ruta absoluta es /home/nombredeusuario.

2. comando cd

Para navegar por los archivos y directorios de Linux, usa el comando **cd**. Te pedirá la ruta completa o el nombre del directorio, dependiendo del directorio de trabajo actual en el que te encuentres. Supongamos que estás en **/home/nombredeusuario/Documentos** y deseas ir a **Fotos**, un subdirectorio de **Documentos**. Para hacerlo, simplemente escribe el siguiente comando: **cd Fotos**. Otro escenario es si deseas ir a un directorio completamente nuevo, por ejemplo, **/home/nombredeusuario/Peliculas**. En este caso, debes escribir **cd** seguido de la ruta absoluta del directorio: **cd /home/ nombredeusuario/Peliculas**.
Hay algunos atajos para ayudarte a navegar rápidamente:

- **cd ..** (con dos puntos) para ir un directorio hacia arriba
- **cd** para ir directamente a la carpeta de inicio
- **cd-** (con un guión) para ir al directorio anterior

Como nota al margen, el shell de Linux distingue entre mayúsculas y minúsculas. Por lo tanto, debes escribir el nombre del directorio de forma exacta.

3. comando ls

El comando ls se usa para ver el contenido de un directorio. Por defecto, este comando mostrará el contenido de tu directorio de trabajo actual. Si deseas ver el contenido de otros directorios, escribe ls y luego la ruta del directorio. Por ejemplo, ingresa ls/home/nombredeusuario/Documentos para ver el contenido de Documentos.

Hay variaciones que puedes usar con el comando ls:

- ls -R también listará todos los archivos en los subdirectorios
- ls -a mostrará los archivos ocultos
- ls -al listará los archivos y directorios con información detallada como los permisos, el tamaño, el propietario, etc.

4. comando cat

cat (abreviatura de concaténate, en inglés) es uno de los comandos más utilizados en Linux. Se utiliza para listar el contenido de un archivo en la salida estándar (sdout). Para ejecutar este comando, escribe cat seguido del nombre del archivo y su extensión. Por ejemplo: cat archivo.txt.

Aquí hay otras formas de usar el comando cat:

- cat > nombredearchivo crea un nuevo archivo.
- cat nombredearchivo1 nombredearchivo2>nombredearchivo3 une dos archivos (1 y 2) y almacena la salida de ellos en un nuevo archivo (3)
- convertir un archivo a mayúsculas o minúsculas, cat nombredearchivo | tr a-z A-Z> salida.txt

5. comando cp

Usa el comando **cp** para copiar archivos del directorio actual a un directorio diferente. Por ejemplo, el comando **cp escenario.jpg /home/nombredeusuario/Imagenes** crearía una copia de **escenario.jpg** (desde tu directorio actual) en el directorio de **Imagenes**.

6. comando mv

El uso principal del comando **mv** es mover archivos, aunque también se puede usar para cambiar el nombre de los archivos.
Los argumentos en mv son similares al comando cp. Debes escribir **mv**, el nombre del archivo y el directorio destino. Por ejemplo: **mv archivo.txt /home/nombredeusuario/Documentos**.
Para cambiar el nombre de los archivos, el comando de Linux es **mv nombreviejo.ext nombrenuevo.ext**

7. comando mkdir

Usa el comando **mkdir** para crear un nuevo directorio: si escribes **mkdir Musica**, creará un directorio llamado **Musica**.
También hay comandos adicionales de **mkdir**:

- Para generar un nuevo directorio dentro de otro directorio, usa este comando básico de Linux **mkdir Musica/Nuevoarchivo**
- Usa la opción **p** (padres) para crear un directorio entre dos directorios existentes. Por ejemplo, **mkdir -p Musica/2020/Nuevoarchivo** creará el nuevo archivo «2020».

8. comando **rmdir**

Si necesitas eliminar un directorio, usa el comando **rmdir**. Sin embargo, rmdir solo te permite eliminar directorios vacíos.

9. comando **rm**

El comando **rm** se usa para eliminar directorios y el contenido dentro de ellos. Si solo deseas eliminar el directorio, como alternativa a rmdir, usa **rm -r**.
Nota: Ten mucho cuidado con este comando y verifica en qué directorio te encuentras. Este comando elimina todo y no se puede deshacer.

10. comando **touch**

El comando **touch** te permite crear un nuevo archivo en blanco a través de la línea de comando de Linux. Como ejemplo, ingresa touch **/home/nombredeusuario/Documentos/Web.html** para crear un archivo HTML titulado **Web** en el directorio **Documentos**.

11. comando **locate**

Puedes usar este comando para **localizar** un archivo, al igual que el comando de búsqueda en Windows. Además, el uso del argumento **-i** junto con este comando hará que no distinga entre mayúsculas y minúsculas, por lo que puedes buscar un archivo incluso si no recuerdas su nombre exacto.
Para buscar un archivo que contenga dos o más palabras, usa un asterisco **(*)**. Por ejemplo, el comando **locate -i escuela*nota** buscará cualquier archivo que contenga la palabra «escuela» y «nota», ya sea en mayúsculas o minúsculas.

12. comando find

Similar al comando **locate**, usando **find** también buscas archivos y directorios. La diferencia es que usas el comando **find** para ubicar archivos dentro de un directorio dado.
Como ejemplo, el comando find **/home/ -name notas.txt** buscará un archivo llamado **notas.txt** dentro del directorio de inicio y sus subdirectorios.
Otras variaciones al usar **find** son:

- Para buscar archivos en el directorio actual, **find . -name notas.txt**
- Para buscar directorios, **/ -type d -name notes.txt**

13. comando grep

Otro comando básico de Linux que sin duda es útil para el uso diario es **grep**. Te permite buscar a través de todo el texto en un archivo dado. Para ilustrar**, grep azul notepad.txt** buscará la palabra azul en el archivo del bloc de notas. Las líneas que contienen la palabra buscada se mostrarán.

14. comando sudo

Abreviatura de «**SuperUser Do**» (SuperUsuario hace), este comando te permite realizar tareas que requieren permisos administrativos o raíz. Sin embargo, no es aconsejable usar este comando para el uso diario, ya que podría ser fácil que ocurra un error si haces algo mal.

15. comando df

Usa el comando **df** para obtener un informe sobre el uso del espacio en disco del sistema, que se muestra en porcentaje y KB. Si deseas ver el informe en megabytes, escribe **df -m**.

16. comando du

Si deseas verificar cuánto espacio ocupa un archivo o un directorio, el comando **du** (Uso del disco, en inglés) es la respuesta. Sin embargo, el resumen de uso del disco mostrará números de bloque de disco en lugar del formato de tamaño habitual. Si deseas verlo en bytes, kilobytes y megabytes, agrega el argumento **-h** a la línea de comando.

17. comando head

El comando **head** se usa para ver las primeras líneas de cualquier archivo de texto. De manera predeterminada, mostrará las primeras diez líneas, pero puedes cambiar este número a tu gusto. Por ejemplo, si solo deseas mostrar las primeras cinco líneas, escribe **head -n 5 nombredearchivo.ext**.

18. comando tail

Este tiene una función similar al comando head, pero en lugar de mostrar las primeras líneas, el comando **tail** mostrará las últimas diez líneas de un archivo de texto. Por ejemplo, **tail -n nombredearchivo.ext**.

19. comando diff

Para abreviar diferencia, el comando **diff** compara el contenido de dos archivos línea por línea. Después de analizar los archivos, genera las líneas que no coinciden. Los programadores a menudo usan este comando cuando necesitan hacer modificaciones al programa en lugar de reescribir todo el código fuente.
La forma más simple de usar este comando es **diff archivo1.ext archivo2.ext**

20. comando tar

El comando **tar** es el comando más utilizado para guardar múltiples archivos en un **tarball**, un formato de archivo de Linux común que es similar al formato zip, con compresión opcional.
Este comando es bastante complejo con una larga lista de funciones, como agregar nuevos archivos a un archivo existente, enumerar el contenido de un archivo, extraer el contenido de un archivo y muchos más.

21. comando chmod

chmod es otro comando de Linux, utilizado para cambiar los permisos de lectura, escritura y ejecución de archivos y directorios. Como este comando es bastante complicado, puedes leer el **tutorial completo** (en inglés) para ejecutarlo correctamente.

22. comando chown

En Linux, todos los archivos son propiedad de un usuario específico. El comando **chown** te permite cambiar o transferir la propiedad de un archivo al nombre de usuario especificado. Por ejemplo, **chown usuariolinux2 archivo.ext** hará que **usuariolinux2** sea el propietario del **archivo.ext**.

23. comando Jobs

El comando **jobs** mostrará todos los trabajos actuales junto con sus estados. Un trabajo es básicamente un proceso iniciado por el shell.

24. comando kill

Si tienes un programa que no responde, puedes cerrarlo manualmente utilizando el comando kill. Enviará una cierta señal al programa que se está ejecutando mal y le indica a la aplicación que finalice.
Hay un total de sesenta y cuatro señales que puedes usar, pero las personas generalmente solo usan dos señales:

- SIGTERM (15): solicita que un programa deje de ejecutarse y te da algo de tiempo para guardar todo tu progreso. Si no especificas la señal al ingresar el comando kill, se utilizará esta señal.
- SIGKILL (9): obliga a los programas a detenerse inmediatamente. El progreso no guardado se perderá.

Además de conocer las señales, también debes conocer el número de identificación del proceso (PID) del programa que deseas detener (kill). Si no conoces el PID, simplemente ejecute el comando ps ux.
Después de saber qué señal deseas usar y el PID del programa, ingresa la siguiente sintaxis:
kill [opción de señal] PID.

25. comando ping

Usa el comando ping para verificar tu estado de conectividad a un servidor. Por ejemplo, simplemente ingresando ping google.com, el comando verificará si puedes conectarte a Google y también medirá el tiempo de respuesta.

26. comando wget

La línea de comandos de Linux es muy útil: incluso puedes descargar archivos de Internet con la ayuda del comando wget. Para hacerlo, simplemente escribe wget seguido del enlace de descarga.

27. comando uname

El comando **uname**, abreviatura de Nombre de Unix, imprimirá información detallada sobre tu sistema Linux, como el nombre de la máquina, el sistema operativo, el núcleo, etc.

28. comando top

Como un terminal equivalente al Administrador de tareas en Windows, el comando **top** mostrará una lista de los procesos en ejecución y la cantidad de CPU que utiliza cada proceso. Es muy útil monitorear el uso de los recursos del sistema, especialmente para saber qué proceso debe terminarse porque consume demasiados recursos.

29. comando history

Cuando hayas estado utilizando Linux durante un cierto período de tiempo, notarás rápidamente que puedes ejecutar cientos de comandos todos los días. Como tal, ejecutar el comando **history** es particularmente útil si deseas revisar los comandos que ingresaste anteriormente.

30. comando man

¿Confundido sobre la función de ciertos comandos de Linux? No te preocupes, puedes aprender fácilmente cómo usarlos directamente desde el shell de Linux mediante el comando **man**. Por ejemplo, al ingresar **man tail** se mostrarán las instrucciones manuales del comando tail.

31. comando **echo**

Este comando se usa para mover algunos datos a un archivo. Por ejemplo, si deseas agregar el texto «Hola, mi nombre es John» en un archivo llamado nombre.txt, debes escribir **echo Hola, mi nombre es John >> nombre.txt**

32. comando zip, **unzip**

Usa el comando **zip** para comprimir tus archivos en un archivo zip y use el comando **unzip** para extraer los archivos comprimidos de un archivo zip.

33. comando **hostname**

Si deseas conocer el nombre de tu host/red, simplemente escribe **hostname**. Agregar un **-I** al final mostrará la dirección IP de tu red.

34. comando **useradd, userdel**

Dado que Linux es un sistema multiusuario, esto significa que más de una persona puede interactuar con el mismo sistema al mismo tiempo. **useradd** se usa para crear un nuevo usuario, mientras que **passwd** agrega una contraseña a la cuenta de ese usuario. Para agregar una nueva persona llamada John, escribe **useradd John** y luego para agregar su contraseña, escribe **passwd 123456789**.
Eliminar un usuario es muy similar a agregar un nuevo usuario. Para eliminar la cuenta de usuario, escribe **userdel NombredeUsuario**
Consejos y trucos extra, usa el comando **clear** para limpiar el terminal si se está abarrotando con demasiados comandos pasados.

Prueba el botón TAB para autocompletar lo que estás escribiendo. Por ejemplo, si necesitas escribir Documentos, comienza a escribir un comando (vamos con cd Docu, luego presione la tecla TAB) y el terminal completará el resto, mostrándote cd Documentos.
Ctrl+C y Ctrl+Z se utilizan para detener cualquier comando que esté funcionando actualmente. Ctrl+C detendrá el comando de forma segura, mientras que Ctrl+Z forzará la detención.
Si accidentalmente congelas tu terminal usando Ctrl+S, simplemente debes deshacer esto con el comando descongelar Ctrl+Q.
Ctrl+A te mueve al comienzo de la línea mientras que Ctrl+E te mueve al final.
Puedes ejecutar varios comandos en un solo comando utilizando «;» para separarlos. Por ejemplo, Comando1; Comando2; Comando 3. O usa && si solo deseas que se ejecute el siguiente comando cuando el primero sea exitoso.

Los comandos básicos de Linux ayudan a los usuarios a ejecutar tareas de manera fácil y efectiva. Puede llevar un tiempo recordar algunos de los comandos básicos, pero nada es imposible con mucha práctica. Al final, conocer y dominar estos comandos básicos de Linux sin duda será beneficioso para ti.