* **Trabajo Práctico : GNU/Linux**
* Fecha de entrega: 19/9/24
* Alumnos:
* Ella Rohana Venice Lo Re Mansilla, 28232

Miguel Angel Lardo, 29812

German Romeo Belli, 31611

Luz Alejandra Rodriguez Salinas, 30569

[Objetivos del trabajo: 7](#_h09lcb4f98rc)

[**Reseña: 8**](#_tqi94jp0h0yv)

[**1. INTRODUCCIÓN 9**](#_w83u15lp8krt)

[1.1. ¿Qué es la cuenta de superusuario (root) y para qué se utiliza? 9](#_v8tv2tivg76b)

[1.2. Ingresar al sistema como superusuario (root), y realizar los siguientes pasos: 9](#_y0ml1wxo4zjf)

[1.3. adduser <apellido> (reemplazar <apellido> por el suyo). 9](#_1tdnqb6qvxa0)

[1.4. passwd <apellido> (Ingrese una contraseña (password) a su elección). 10](#_j5po9lnlgvau)

[1.5. logout 10](#_o4lri9q2d180)

[1.6. Indique claramente qué efectuaron estos comandos, e indique qué archivo/s fueron modificados (Dentro del directorio /etc). 11](#_f3w0ryvd51r8)

[1.7. ¿En qué directorio se encuentran los comandos utilizados en los puntos 1.3 y 1.4? 11](#_8jzcijkxjguf)

[**2. TECLADO / TERMINALES 12**](#_jdugnliueaol)

[2.1. ¿Qué sucede si tecleo cat /e <tab> p <tab>? (donde tab es la tecla tabulación). Presione <tab> nuevamente ¿Qué pasó ahora? 12](#_a0wg8667g694)

[2.2. ¿Qué sucede si tecleo cat /e <tab> pas <tab>? 12](#_es5ye22egnd)

[2.3. En este punto analizaremos las distintas terminales que hay en un sistema GNU/Linux. Ejecute los siguientes comandos e indique cuál fue el resultado: 12](#_wkra0eg4trtj)

[2.3.1. who 12](#_699f2boalzun)

[2.3.2. Presione las teclas <ctrl> y <alt>, y sin soltarlas presione cualquiera de las teclas de función (de F3 a F6 para Ubuntu). En la pantalla debería aparecer el login del sistema, de lo contrario, ejecute el paso nuevamente presionando otra tecla de función. Si ya tiene el login del sistema vuelva a conectarse. 13](#_mv416pvnp7rx)

[2.3.3. Ejecute nuevamente el comando who. ¿Qué diferencias encuentra con la primera vez que lo ejecutó? 13](#_wpt0r7ag7iwl)

[2.3.4. Ejecute el comando who am i ¿qué muestra?, ¿Qué diferencias tiene con el comando ejecutado en el punto anterior? 14](#_spqonny0yu7h)

[2.3.5. Repita el paso 2.3.2 y el 2.3.3 hasta que no encuentre ninguna sesión para abrir. 14](#_99mifdcw5i6l)

[2.3.6. Una vez terminado el punto anterior, Ud. se encontrará sesionado en el sistema como mínimo cinco veces. Lo que acaba de hacer es abrir cuatro terminales virtuales (que podrían ser usadas por distintos usuarios, con diferentes perfiles), en la misma máquina. Así como existen terminales virtuales dentro del mismo equipo, si Ud. cuenta con una red, o con terminales tipo serie, podría abrir tantas sesiones de trabajo como Ud. quiera o necesite. Investigue e indique cómo se denominan los distintos tipos de terminales, y cuáles son los archivos que las representan (tip: busque en el directorio /dev). 14](#_7nhp9odcepsi)

[2.3.7. Use el comando logout para cerrar las sesiones abiertas en todas las terminales virtuales y vuelva a la sesión gráfica (<ctrl> <alt> F2 para Ubuntu). 15](#_tjj1v9t5ooj9)

[**3. AYUDA 16**](#_z48py7rhlyyk)

[3.1. INFO: Info es un programa para leer documentación. Este se compone de una estructura del tipo árbol, dividido en nodos de información. Cada nodo describe un específico tópico con un determinado nivel de detalle. 16](#_50mt2nhh11j1)

[3.1.1. Ingrese a info y responda: 16](#_z4d9bjobjpbo)

[3.1.1.1. ¿Cómo se llama el nodo raíz de Info? 16](#_x7qu6g90i8p5)

[3.1.1.2. Ubique el cursor en la línea (\* cp:) y presione ENTER. 16](#_kqi7jycfnn68)

[3.1.1.3. ¿Qué sucedió? 17](#_7c8ayevttvlq)

[3.1.1.4. ¿Cómo se llama este nodo? 18](#_h2gu6d8vekv6)

[3.1.1.5. ¿Cuál es el próximo nodo? 18](#_4x3i8rr2559j)

[3.1.1.6. ¿Cómo puedo moverme al próximo nodo? 18](#_pqzxkxc25wx7)

[3.1.1.7. ¿Cómo puedo moverme al nodo anterior? 18](#_muyqsl789snh)

[3.1.2. Presione la tecla 'u'. 18](#_41cyabmtouwa)

[3.1.2.1. ¿Qué sucedió? 19](#_x2614nftm2qg)

[3.1.2.2. ¿En qué nodo se encuentra? 19](#_kdh2x2m7pk1e)

[3.1.3. Repita el punto 3.1.2. hasta que llegue a la raíz de Info. 19](#_n8k2ms6dda1v)

[3.1.3.1. ¿Con qué tecla puedo volver directamente a este nodo?. 19](#_z40a2o8jfras)

[3.1.3.2. ¿Cuál es el método directo para acceder al nodo cp? (tip: sin desplazar el cursor). 19](#_7exqsf1unx4q)

[3.1.4. ¿Cómo puedo buscar una palabra clave dentro de un nodo? 19](#_f2jrj419qwf8)

[3.1.5. ¿Cómo puedo buscar la siguiente palabra clave, buscada anteriormente? 20](#_qp93uor54t5j)

[3.1.6. ¿Cómo puedo salir de Info? -hágalo. 20](#_s7tyx6yj0h2d)

[3.2. MAN: man es un programa que formatea y muestra las páginas del manual. 20](#_ba6r81orrcvs)

[3.2.1. ¿Cuál es la diferencia entre man e info? 20](#_wu1l6gqyg8y3)

[3.2.2. ¿Cómo puedo ver la información de un determinado comando? 21](#_jskcet5owuaz)

[3.2.3. ¿Cómo puedo buscar una palabra clave dentro de la página del manual? 22](#_oi26t7r6no26)

[3.2.4. ¿Cómo puedo salir? 22](#_z7lb6xiks6me)

[3.2.5. ¿Cómo hago para buscar una palabra clave determinada en todas las páginas del manual? 22](#_kb9xzx78wkdj)

[3.2.6. ¿Qué es lo sucede al realizar lo siguiente? 23](#_zhw00c67dgly)

[3.2.6.1. man 23](#_j9qqhvldcp87)

[3.2.6.2. man man 23](#_ax5qlkolkw5r)

[3.2.6.3. man cp 23](#_r70kwfdprg4y)

[3.2.6.4. man printf 24](#_y7g0su4jkzsd)

[3.2.6.5. man fprintf 25](#_bzpy2s5m5lbg)

[3.2.6.6. man sprintf 25](#_okvd9jrn7fzm)

[3.2.6.7. man 3 printf 26](#_slno75nc37o)

[3.1.2. Del punto anterior, responder: 26](#_d4v28jo4d1qa)

[3.1.2.1. Al invocar man junto con fprintf y sprintf muestra la misma página. ¿Porqué no muestra la misma página al invocarlo con printf?. (TIP: vea el punto 3.2.6.2). 26](#_e5fdykxivkg2)

[3.1.2.2. ¿Cómo puedo invocar al man para ver directamente la función printf del lenguaje C?. 27](#_3iv4ana55g3j)

[3.2. HELP: help es la ayuda que ofrece el shell de GNU/LINUX para utilizar sus comandos. 27](#_do2c07oops7x)

[3.2.1. ¿Cuál es la diferencia entre help e info?. 27](#_79qst4pzecr6)

[3.2.2. ¿Cuál es la diferencial entre help y man?. 28](#_3oh6ck7c74l9)

[3.2.3. ¿Qué sucede al invocar al help?. 29](#_wlltymb10oem)

[3.2.4. ¿Cómo puedo ver la información de un determinado comando? 29](#_esrs3q6uy301)

[3.2.5. ¿Qué sucede al utilizar el comando help cd? 29](#_70ucl8o9zumw)

[3.3. whereis 30](#_e941p3jgjhfz)

[3.3.1. ¿Qué sucede al utilizar el comando whereis cp? 30](#_99pksrv0mz6z)

[3.3.2. ¿Qué es la información que se muestra por pantalla al ejecutar el punto anterior? 30](#_3yvmhcfla8a2)

[3.3.3. ¿Qué ocurre si se ejecuta whereis \* sobre un directorio? (Tip: si no pasa nada, intentalo nuevamente pero primero ejecute cd /bin) 30](#_a5cy63s109km)

[3.3.4. ¿Cuál es la diferencia entre whereis y find? 30](#_j9rqbss2vaj6)

[3.4. whatis 31](#_z72b0nl4vqpa)

[3.4.1. ¿Qué sucede al utilizar el comando whatis cp? 31](#_4q8rjkxig83y)

[3.4.2. Cambie al directorio /bin, y ejecute el comando whatis \* ¿Qué ocurrió? 31](#_qaf1pcn71et3)

[3.4.3. Utilice el comando apropos passwd y whatis passwd. Enumere las diferencias encontradas en el resultado de cada uno de los comandos. 31](#_z36q1exiemxh)

[**4. DIRECTORIOS 33**](#_ap1clwvtgwi7)

[4.1. ¿Para qué se usa el comando cd? Ejecute las siguientes variantes de cd e indique cuál fue el resultado obtenido: 33](#_5xrtj64md5nm)

[4.1.1. cd / 33](#_4tob3c24z0kh)

[4.1.2. cd 33](#_79zrc5m9ippa)

[4.1.3. cd /etc 34](#_wwaqlsy8mjpm)

[4.1.4. cd.. 34](#_xs0rtraragu3)

[4.1.5. cd .. 34](#_byd52fkanoa)

[4.2. Bash sobre directorios: 34](#_2mcflhx2vr7d)

[4.2.1. ¿Cuáles son las diferencias entre el path absoluto y el path relativo? 34](#_u4sakb43t5s5)

[4.2.2.1. cd ~ 35](#_q3qwe3wx7zq4)

[4.2.2.2-cd – 35](#_6hf2gx60ej5i)

[4.2.3 ¿Cuál es la diferencia entre cd ~ y cd -? 35](#_sq2neh1x6mnp)

[4.3 Operaciones con directorios: 36](#_13ya2fw8fvfr)

[4.3.1 ¿Con que comando se puede crear un directorio? 36](#_qzmdyqgw26td)

[4.3.2 ¿Con qué comando se puede borrar un directorio? 36](#_m33gtuwx1lml)

[4.3.3¿Qué sucede sí el directorio no está vacío? 36](#_rwgal743ahpa)

[4.3.4¿Cómo puedo salvar la situación anterior? (Sin borrar uno a uno los archivos existentes). 37](#_v4c39u3kjfsb)

[4.4. ¿Qué significa la expresión ./ cuando se utiliza delante de un archivo? ¿Para que sirve? 37](#_1m1soncrn653)

[4.5. ¿Cómo puede moverse entre directorios sin utilizar el PATH completo? 37](#_jz2luze8a9p9)

[5. ARCHIVOS 38](#_echinow06j0z)

[5.1¿Qué hacen los siguientes comandos? 38](#_q0iixnzcjcs6)

[5.1.1. Cp 38](#_d2d9vrvvfbj5)

[5.1.2. mv 38](#_p8398neyn40q)

[5.1.3.rm 39](#_yx5rfqw9amz1)

[5.1.4 rcp 39](#_iilsvzvcnm56)

[5.1.5. rsh 39](#_ejyaaxdxtie5)

[5.1.6 scp 39](#_spdcxlfuzaa5)

[5.1.7. ssh 40](#_62lhs43z77z0)

[5.2. ¿Con qué comando puedo concatenar el contenido de dos archivos? 40](#_7yl0lxyhcdyv)

[5.2.1. ¿Se puede usar ese comando para otra cosa? 40](#_jbwh9iut4ru4)

[5.3 Haga un ls –l /dev 41](#_usieeug5b6fi)

[5.3.1 ¿Qué significa el primer carácter? 41](#_blyew7x2ybs)

[5.3.2. ¿Cuáles son todos los posibles valores que puede contener ese campo y que 41](#_f9qc20bxsnw3)

[5.4 ¿Para qué sirve el comando touch? ¿qué utilidad le encuentra? 42](#_dlvqeaufbrxp)

[6.PERMISOS 43](#_8qh2911mzva0)

[6.1. Teniendo en cuenta el ls –l anterior, ¿indique que son los siguientes 9 caracteres? (sin considerar el primero sobre el que ya respondió anteriormente) 43](#_bu9ufx9bnrbj)

[6.2. ¿qué significan cada caracter? ¿cómo están agrupados? 43](#_idquol632oje)

[6.3. ¿Cómo se asignan los permisos? (detalle los comandos). 43](#_utaaz5kexr9r)

[6.4. ¿Qué son el owner, y el group de un archivo? ¿Se pueden cambiar? 45](#_izglgunc1okd)

[6.5. Intente cambiar los permisos de un archivo perteneciente al root (sesionado como usuario). Explique qué sucedió. 45](#_4z5qgltzo9qy)

[6.6. Explique la forma de cambiar los permisos con valores en octal. 45](#_598tlcrt59o8)

[6.7. ¿Cuál es el significado de los permisos en los directorios (se debe indicar que indica una r, una w, y una x)? 46](#_lm92r3wivbzo)

[7.FILTROS 47](#_7bfxpt8e62he)

[7.1.¿Cuál es la diferencia de los comandos more, less y cat? De un ejemplo de cada uno. 47](#_3snyc2fpm2sf)

[7.2.¿Cuál es la diferencia entre tail y head? 47](#_guqhe3ve78ku)

[7.3.¿Qué es lo que realiza el comando uniq? 47](#_nbpu4c1fchx3)

[7.4.¿Qué es lo que realiza el comando grep? 47](#_qmne5jrz0qhp)

[**8. VI 48**](#_phcg6h2ykbib)

[8.1. Ejecute la siguiente instrucción: vi $HOME/prueba.txt ¿Qué sucedió? Ahora ejecute todos los pasos detallados a continuación. 48](#_fj464q598i)

[8.1.1. Escriba la siguiente frase: “Este es el archivo prueba.txt de <nombre y apellido>” 48](#_pfn5913r8jyn)

[8.1.2. ¿Qué tuvo que hacer para poder escribir la frase? 48](#_cpr88gy2r7z2)

[8.1.3. Guarde el archivo, y salga del editor. ¿Qué comando utilizó? 49](#_oem3bxv7itoh)

[8.1.4. Ingrese nuevamente al archivo. 49](#_jvt6cbexoem7)

[8.1.5. Incorpore al inicio del archivo el siguiente párrafo (los acentos pueden ser evitados): 49](#_9gb6s2uetnaz)

[8.1.6. Describa todos los pasos que tuvo que realizar. 49](#_f3g0f9wmpm7j)

[8.1.7. Guarde el archivo y continúe la edición. ¿Qué comandos utilizó? 50](#_3v3aro2w4v5j)

[8.1.8. Borre la línea de “Legajo”. Indique por lo menos dos formas de realizarlo. 50](#_1unshcgv2rib)

[8.1.9. Invierta el orden de las líneas “Curso y TP”. No está permitido rescribirlas. ¿Qué comandos utilizó? 50](#_o4jcz84wrr9p)

[8.1.10. Ubíquese en la línea 2 (dos) del archivo. No está permitido usar las teclas del cursor, ni el mouse. ¿Qué comando utilizó? 51](#_suj0y9bosmkb)

[8.1.11. Marque para copiar las líneas 2, 3, y 4 (todas juntas, no de a una a la vez). ¿Cómo lo realizó? 51](#_443sk1xe7h25)

[8.1.12. Ubíquese al final del archivo (sin usar las teclas del cursor), y pegue dos veces el contenido del buffer. ¿Qué comando usó? 51](#_muxvnrcja6re)

[8.1.13. Deshaga uno de los copiados. No está permitido borrar línea por línea, ni carácter a carácter. ¿Qué comando usó? 51](#_k1eq1jay5wtl)

[8.1.14. ¿Cómo busco la palabra “Documento”? ¿Cómo busco la segunda ocurrencia de una palabra? 51](#_sj8h3si986b1)

[8.1.15. ¿Cómo puedo reemplazar la palabra “Documento” por “Documento:” (sin borrar, o realizar el reemplazo a mano)’ 51](#_9pmedg8dxtjt)

[8.1.16. Guarde el archivo y salga. 51](#_5meh5ru5q6lp)

[8.1.17. Ejecutar "vi miprog.c" 51](#_jjpsykx97csh)

[**9. VARIABLES DE ENTORNO 56**](#_3asqlxqys9zu)

[9.1. ¿Qué son las variables de entorno y para qué sirven? 56](#_etnum8dgogoe)

[1. Variable PS1 (Prompt String 1): 58](#_m4bquox7ipw6)

[PS1 define la apariencia del prompt principal en Bash. Es el texto que se ve cuando la terminal está esperando que escribas un comando. 58](#_a7y5z4mnnart)

[2. Variable PS2 (Prompt String 2): 58](#_a1w27x8bcmgh)

[Comando Multilínea: 59](#_bzqr4q308lyl)

[**Conclusiones finales: 59**](#_feyqz9aoqttb)

### **Objetivos del trabajo:**

1. **Entender el Concepto de Superusuario (root) en Linux**:
   * Definir qué es la cuenta de superusuario y su propósito en la administración del sistema.
2. **Realizar Operaciones Básicas como Superusuario**:
   * Ejecutar comandos para agregar y gestionar usuarios y contraseñas.
   * Comprender los cambios realizados en los archivos de configuración del sistema ubicados en el directorio /etc.
3. **Comprender el manejo de los Archivos** :
   * crear archivos
   * crear directorios
   * borrar archivos
   * borrar directorios
   * borrar rutas completas de directorios
   * modificar archivos
4. **Explorar la Utilización de la Tecla Tab para Autocompletado**:
   * Experimentar con la tecla de tabulación para el autocompletado de comandos y nombres de archivos, y entender cómo mejora la eficiencia en la línea de comandos.
5. **Analizar la Gestión de Terminales Virtuales en Linux**:
   * Ejecutar y comparar el comando who para identificar usuarios y sesiones activas.
   * Investigar y comprender el concepto de terminales virtuales y su representación en el directorio /dev.
6. **Cerrar Sesiones y Manejar Múltiples Terminales**:
   * Usar el comando logout para cerrar sesiones y gestionar múltiples terminales virtuales.
7. **Utilizar el Sistema de Documentación de Linux: INFO y MAN**:
   * Navegar por el sistema de ayuda de Linux usando los comandos info y man.
   * Comprender las diferencias entre info y man, y cómo buscar información dentro de las páginas de manual.
8. **Explorar Comandos Adicionales de Ayuda como whereis, whatis, y help**:
   * Investigar la funcionalidad de los comandos whereis y whatis para localizar archivos y obtener descripciones rápidas de comandos.
9. **Manejar Directorios y Rutas en la Línea de Comandos**:
   * Utilizar el comando cd para navegar entre directorios y entender la diferencia entre rutas absolutas y relativas.
   * Comprender el uso de ./ y cómo moverse entre directorios sin especificar la ruta completa.
   * Manejos de comandos adicionales para manipular archivos y directorios.
10. **Aprender a Utilizar el Editor de Texto vi**:
    * Familiarizarse con los modos de operación de vi y cómo realizar operaciones básicas como editar, guardar y salir de archivos.
    * Ejecutar comandos específicos en vi para realizar tareas como copiar, pegar, buscar y reemplazar texto.
11. **Comprender el Concepto y la Utilidad de las Variables de Entorno**:
    * Definir qué son las variables de entorno, cómo se configuran y para qué se utilizan en el contexto de la configuración del entorno de trabajo en Linux.
12. **Manejo Avanzado de la Línea de Comandos**:
    * Aprender a utilizar herramientas avanzadas de línea de comandos como grep, awk, sed, y find.
    * Practicar el uso de redirección, tuberías y comandos compuestos.
13. **Configuración y Gestión de Usuarios y Permisos**:
    * Aprender a gestionar usuarios y grupos, incluyendo la creación, eliminación y modificación.
    * Comprender y aplicar permisos de archivo y control de acceso.
14. **Automatización de Tareas con Scripts**:
    * Aprender a escribir y ejecutar scripts en bash para automatizar tareas comunes.
    * Entender conceptos básicos de programación de scripts, como variables, estructuras de control y funciones.
15. **Documentación y Mejores Prácticas**:
    * Aprender a consultar y utilizar la documentación del sistema, como man y info.
    * Adoptar las mejores prácticas para la documentación y presentación de informes técnicos relacionados con Linux.

## **Reseña:**

En el ámbito de los sistemas operativos, Linux se destaca como una plataforma robusta y versátil, ampliamente utilizada en servidores, estaciones de trabajo y dispositivos embebidos.

Este trabajo tiene como objetivo principal proporcionar una comprensión integral de los aspectos fundamentales del sistema operativo Linux, abarcando desde la administración de usuarios y terminales hasta el manejo avanzado de archivos y la utilización de herramientas de ayuda y edición.

Este trabajo no solo proporcionará una visión técnica detallada de cómo utilizar y administrar Linux, sino que también permitirá desarrollar habilidades prácticas esenciales para la interacción con este sistema operativo.

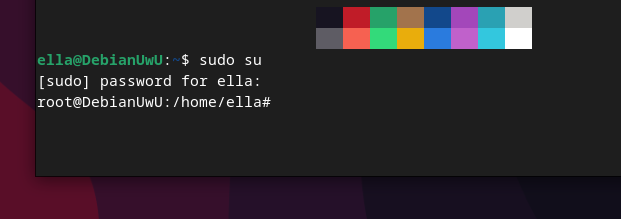
La comprensión de estos conceptos fundamentales facilitará el trabajo en entornos Linux y contribuirá a una mayor competencia en el manejo y administración del sistema.

## **INTRODUCCIÓN**

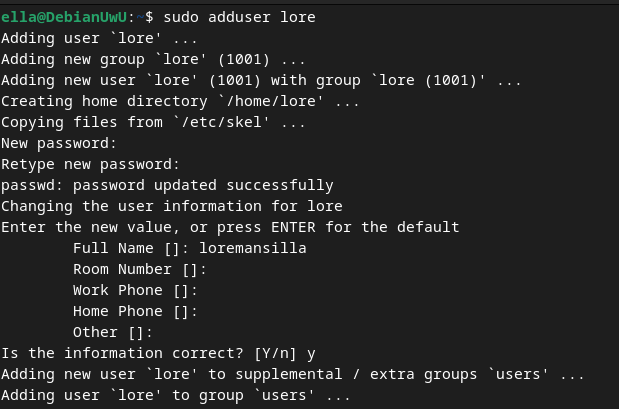
### **1.1.** ¿Qué es la cuenta de superusuario (root) y para qué se utiliza?

La cuenta de superusuario o root, se le denomina a la cuenta con privilegios totales, es decir, tiene permisos para realizar CUALQUIER acción en el sistema operativo.

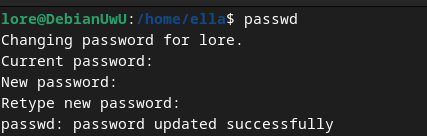
### **1.2.** Ingresar al sistema como superusuario (root), y realizar los siguientes pasos:



### **1.3.** adduser <apellido> (reemplazar <apellido> por el suyo).

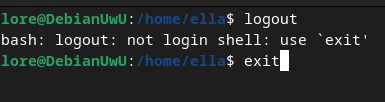


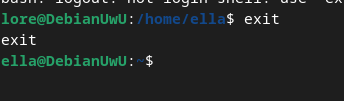
### **1.4.** passwd <apellido> (Ingrese una contraseña (password) a su elección).



### **1.5.** logout

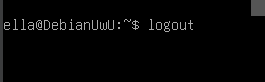
Dado que no nos encontramos en el Login debemos utilizar Exit.

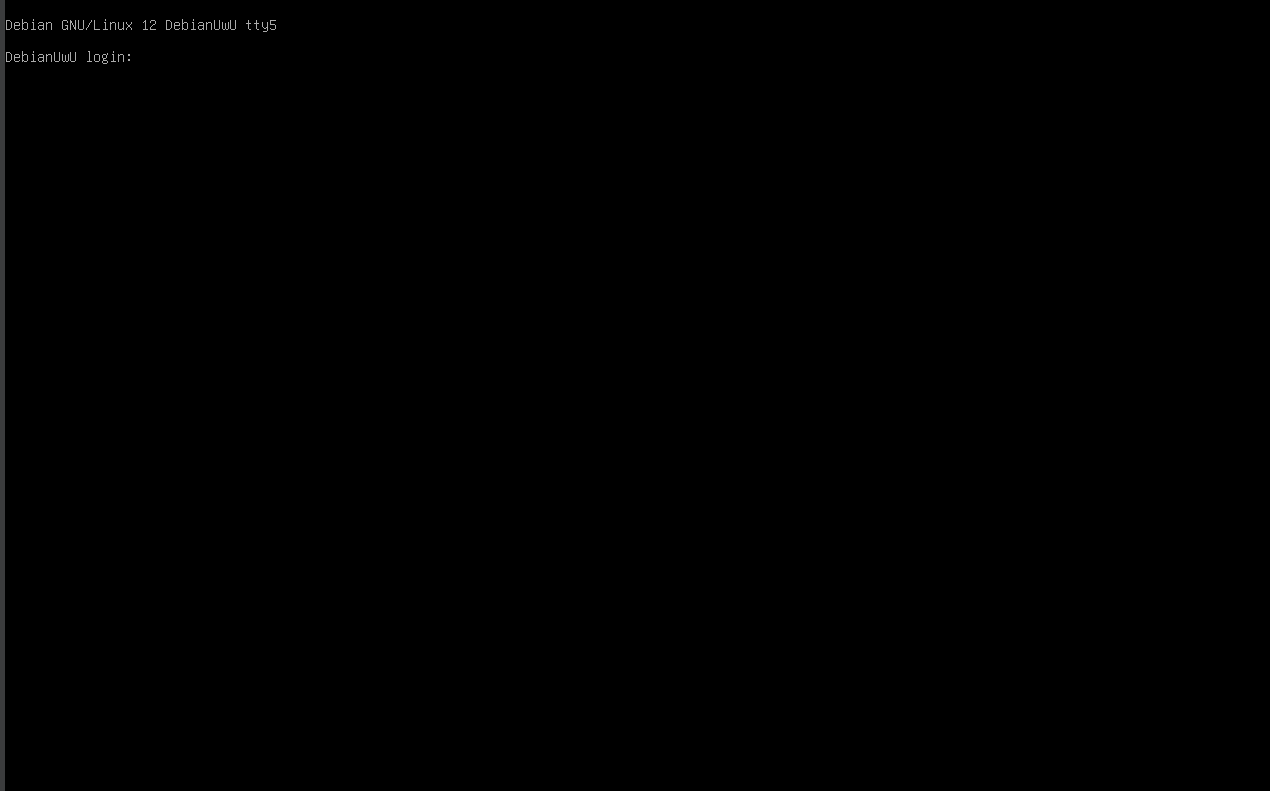




En caso de encontrarnos en el Login:





Luego de ejecutar el comando, la pantalla que visualizamos será la siguiente

### **1.6.** Indique claramente qué efectuaron estos comandos, e indique qué archivo/s fueron modificados (Dentro del directorio /etc).

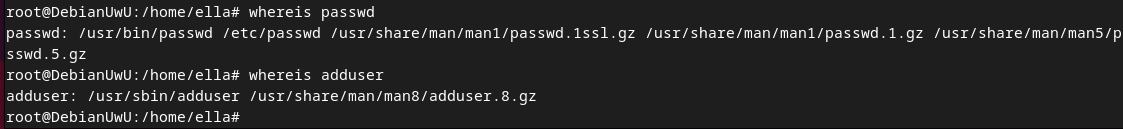
adduser: es un script que facilita la creación de nuevos usuarios, configurando automáticamente varias cosas como el directorio personal, el grupo de usuario y los permisos. Este comando desencadena la creación de una password, creando una nueva entrada para la misma en el directorio **/etc.**

passwd: se utiliza para establecer o cambiar la contraseña del usuario especificado. Actualiza el hash de la contraseña del usuario en el archivo **/etc/shadow**, el mismo contiene hashes de contraseñas de todos los usuarios del sistema.

logout: se utiliza para cerrar la sesión del usuario actual en una terminal de texto o consola. Modifica los archivos relacionados con el registro de sesiones y usuarios, **/etc/utmp** y **/etc/ wtmp**  registran información sobre los inicios y cierres de sesión incluyendo la hora y el origen de la sesión.

### **1.7.** ¿En qué directorio se encuentran los comandos utilizados en los puntos 1.3 y 1.4?

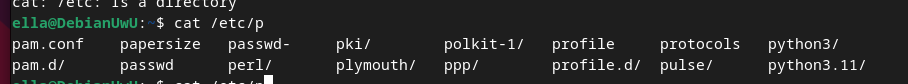
Podemos verificar el directorio de cada comando mediante “whereis”



## **2.** **TECLADO / TERMINALES**

### **2.1.** ¿Qué sucede si tecleo cat /e <tab> p <tab>? (donde tab es la tecla tabulación). Presione <tab> nuevamente ¿Qué pasó ahora?

Autocompleta el “e” con etc, si se clickea nuevamente el sistema nos dará la opción de mostrar todas las opciones de autocompletado que comienzan con la letra indicada, en este caso, “p”.



### **2.2.** ¿Qué sucede si tecleo cat /e <tab> pas <tab>?

Se autocompleta con passwd.



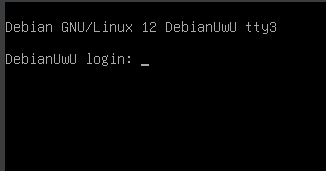
### **2.3.** En este punto analizaremos las distintas terminales que hay en un sistema GNU/Linux. Ejecute los siguientes comandos e indique cuál fue el resultado:

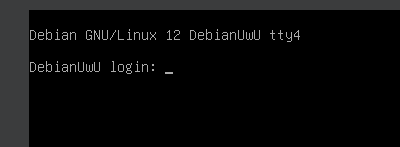
#### **2.3.1.** who

Who muestra el usuario que estoy utilizando, la terminal en la que me encuentro en ese momento y la fecha y hora en la que inicie sesión

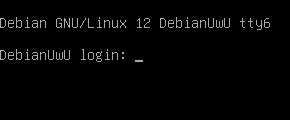


#### **2.3.2.** Presione las teclas <ctrl> y <alt>, y sin soltarlas presione cualquiera de las teclas de función (de F3 a F6 para Ubuntu). En la pantalla debería aparecer el login del sistema, de lo contrario, ejecute el paso nuevamente presionando otra tecla de función. Si ya tiene el login del sistema vuelva a conectarse.









#### **2.3.3.** Ejecute nuevamente el comando who. ¿Qué diferencias encuentra con la primera vez que lo ejecutó?

Aquí se muestra la misma información pero correspondiente a todos los usuarios presentes en el sistema operativo.



#### **2.3.4.** Ejecute el comando who am i ¿qué muestra?, ¿Qué diferencias tiene con el comando ejecutado en el punto anterior?

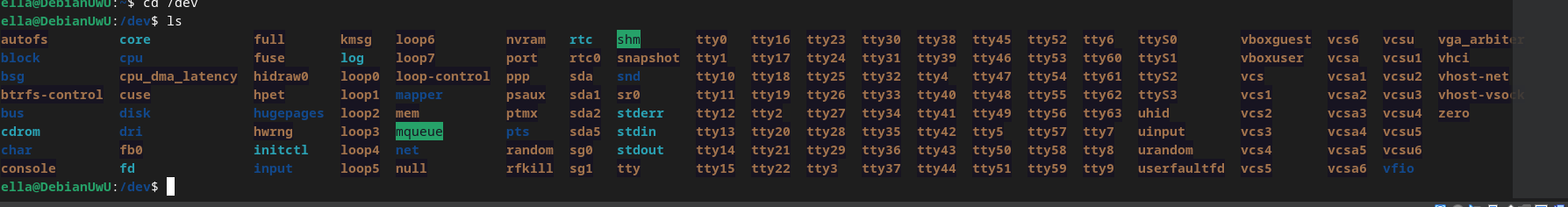
En este caso, la terminal ha cambiado. De estar en tty2 pasamos a estar en tty5.



#### **2.3.5.** Repita el paso 2.3.2 y el 2.3.3 hasta que no encuentre ninguna sesión para abrir.

Se pueden abrir, por teclado, hasta tty6.

#### **2.3.6.** Una vez terminado el punto anterior, Ud. se encontrará sesionado en el sistema como mínimo cinco veces. Lo que acaba de hacer es abrir cuatro terminales virtuales (que podrían ser usadas por distintos usuarios, con diferentes perfiles), en la misma máquina. Así como existen terminales virtuales dentro del mismo equipo, si Ud. cuenta con una red, o con terminales tipo serie, podría abrir tantas sesiones de trabajo como Ud. quiera o necesite. Investigue e indique cómo se denominan los distintos tipos de terminales, y cuáles son los archivos que las representan (tip: busque en el directorio /dev).



**Terminales Virtuales (TTY):**

Se encuentran en /dev/tty\*, son archivos de dispositivo de caracteres. Representan terminales virtuales como las vistas en los puntos anteriores (tty2, tty3, tty4, tty5, etc)

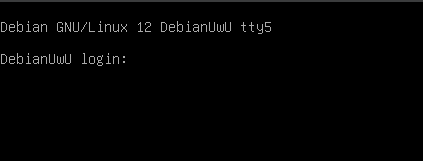
**Terminales de Consola Virtual (VT):**

Se encuentra en /dev/ttyS\*, son archivos de dispositivo de caracteres. Representan puertos serie, que a menudo se usan para conexiones de consola en sistemas antiguos.

#### **2.3.7.** Use el comando logout para cerrar las sesiones abiertas en todas las terminales virtuales y vuelva a la sesión gráfica (<ctrl> <alt> F2 para Ubuntu).



Luego del Logout



## **3.** **AYUDA**

### **3.1.** INFO: Info es un programa para leer documentación. Este se compone de una estructura del tipo árbol, dividido en nodos de información. Cada nodo describe un específico tópico con un determinado nivel de detalle.

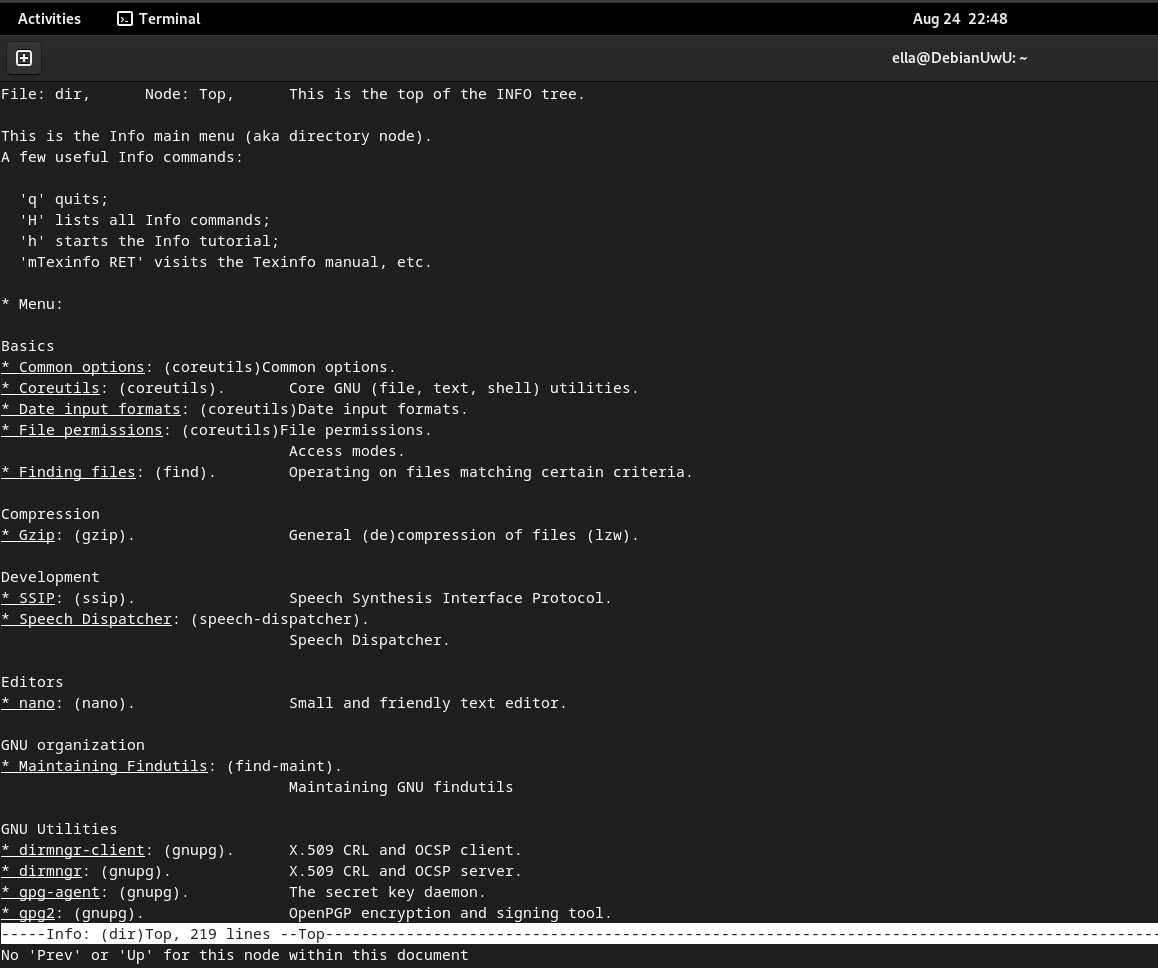
#### **3.1.1.** Ingrese a info y responda:

Ingresamos a info mediante el comando “info”.



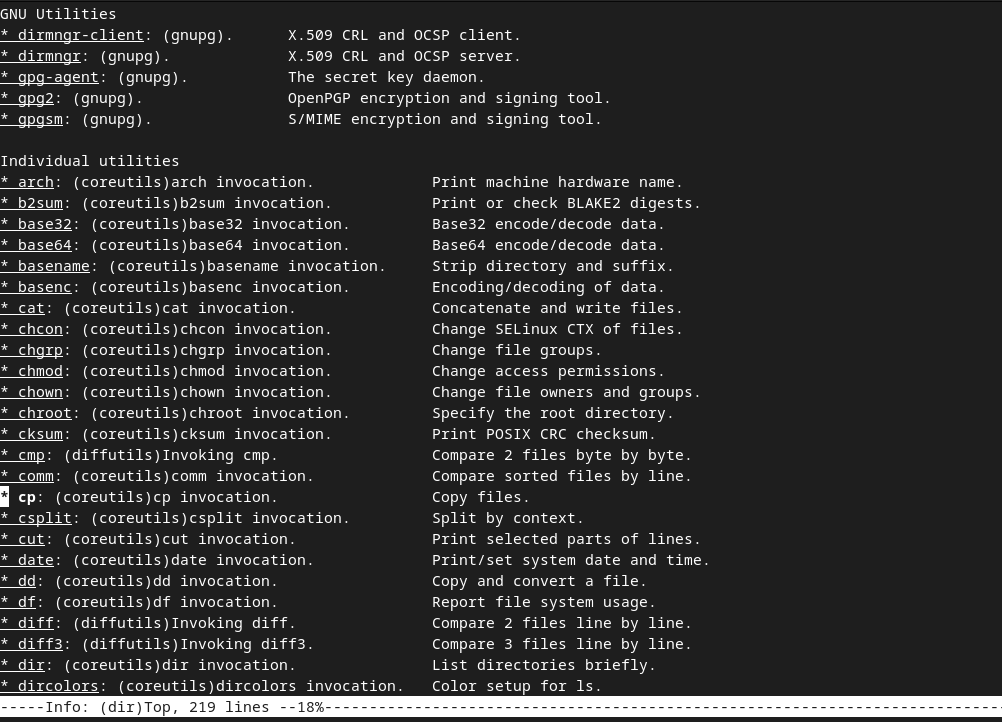
##### **3.1.1.1.** ¿Cómo se llama el nodo raíz de Info?

El nodo raíz se denomina TOP.



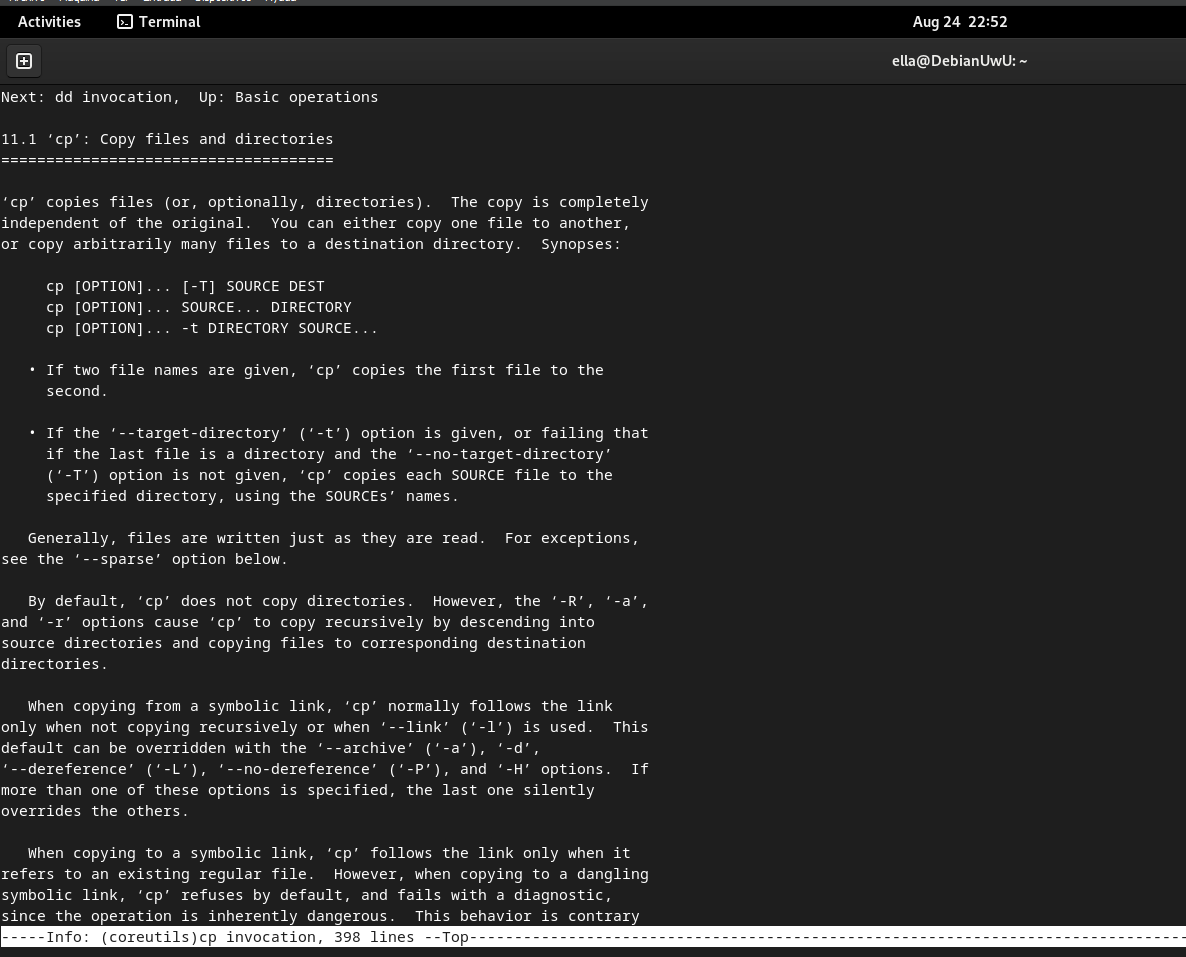
##### **3.1.1.2.** Ubique el cursor en la línea (\* cp:) y presione ENTER.

Nos desplazamos mediante las flechas del teclado.



##### **3.1.1.3.** ¿Qué sucedió?

Se abrió el nodo correspondiente a donde estaba el cursor, en este caso, cp.

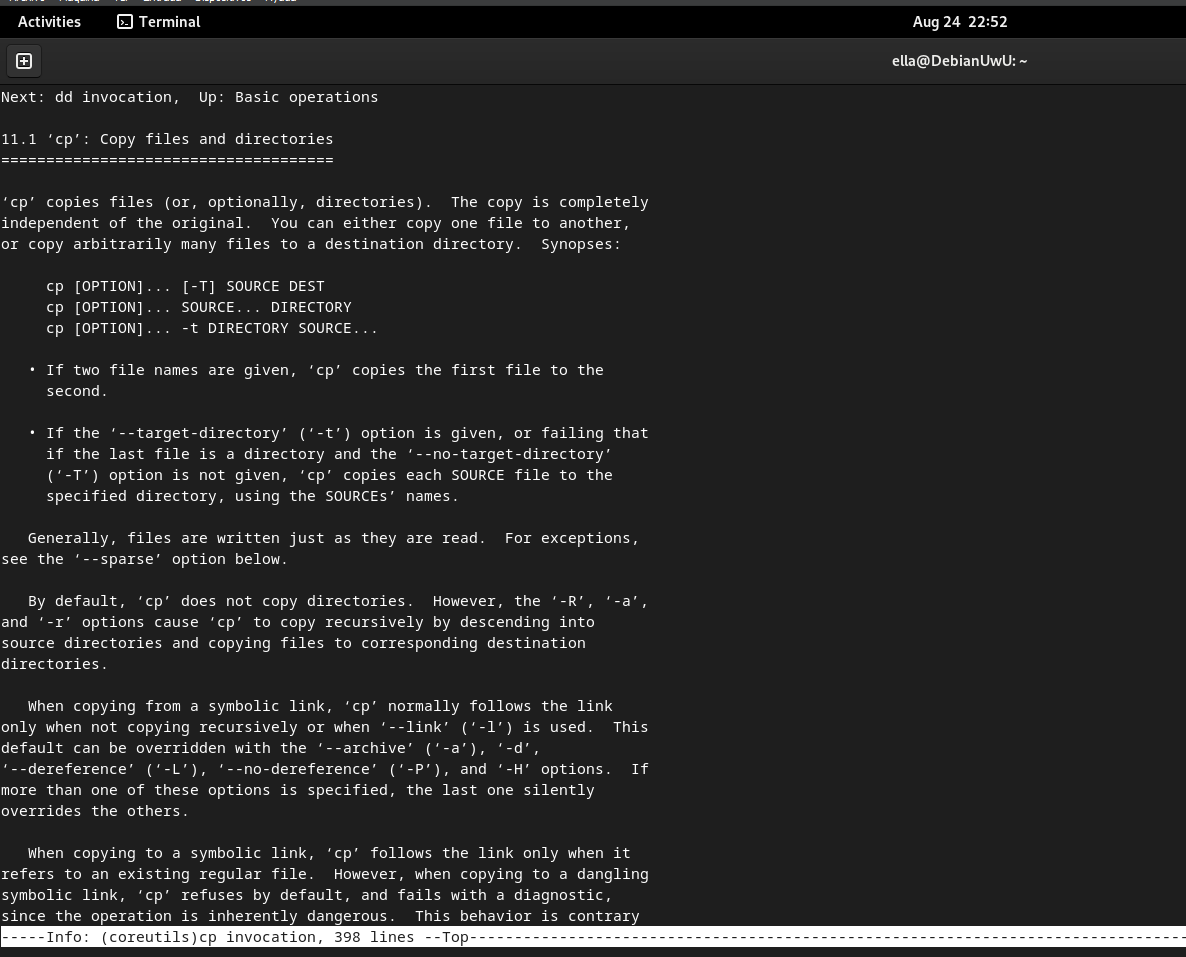


##### **3.1.1.4.** ¿Cómo se llama este nodo?

Este nodo se llama “cp”, corresponde al nodo que ingresamos

##### **3.1.1.5.** ¿Cuál es el próximo nodo?

El nodo siguiente es “dd invocation”.



##### **3.1.1.6.** ¿Cómo puedo moverme al próximo nodo?

Nos podemos desplazar de nodo mediante la letra “n” (Next)

##### **3.1.1.7.** ¿Cómo puedo moverme al nodo anterior?

Nos podemos desplazar de nodo mediante la letra “p” (Previous)

#### **3.1.2.** Presione la tecla 'u'.

La tecla U (Up) nos llevará a una jerarquía superior del nodo en el que nos encontramos.



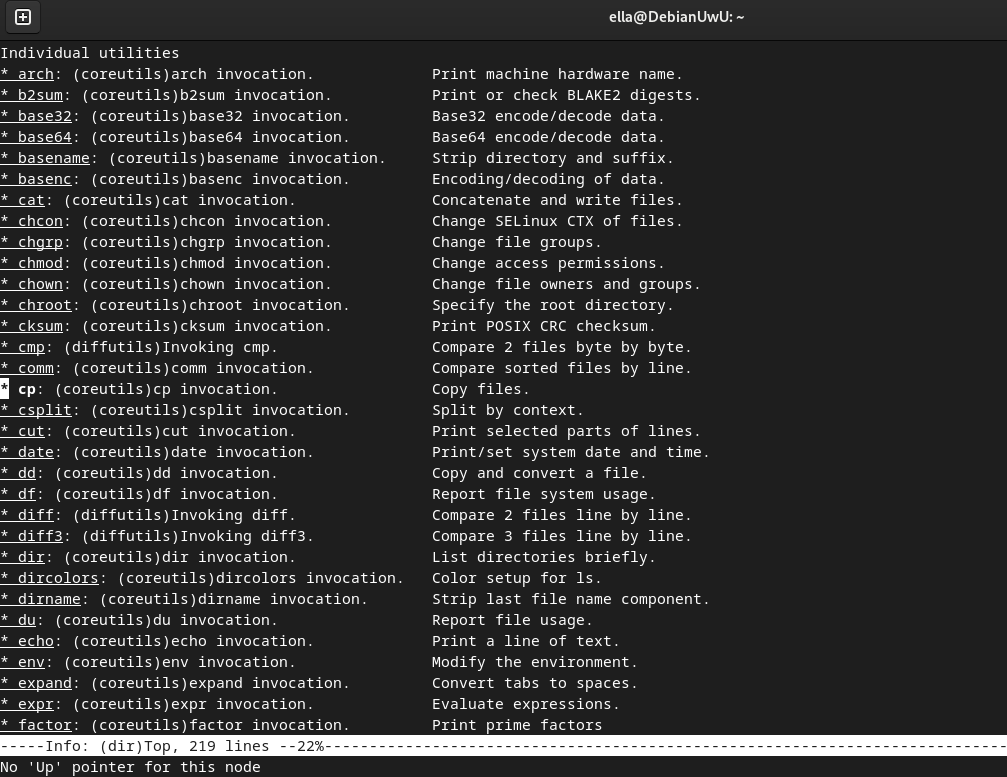
##### **3.1.2.1.** ¿Qué sucedió?

Se muestra en pantalla el contenido que corresponde al nodo.

##### **3.1.2.2.** ¿En qué nodo se encuentra?

Nos encontramos en el nodo de Basic Operations.

#### **3.1.3.** Repita el punto 3.1.2. hasta que llegue a la raíz de Info.



##### **3.1.3.1.** ¿Con qué tecla puedo volver directamente a este nodo?.

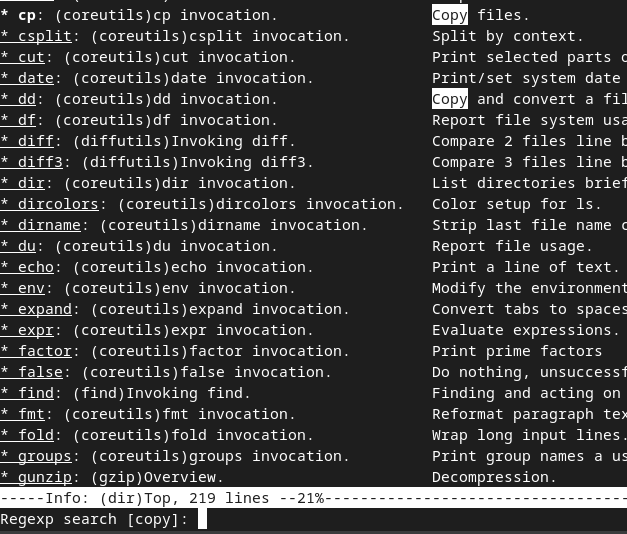
Podemos volver al nodo raiz mediante la letra “T”

##### **3.1.3.2.** ¿Cuál es el método directo para acceder al nodo cp? (tip: sin desplazar el cursor).

Podemos escribir en consola “info cp”, esto abrirá su nodo directamente.



#### **3.1.4.** ¿Cómo puedo buscar una palabra clave dentro de un nodo?

Una vez ingresamos a un nodo podemos buscar una palabra clave mediante la letra S (Search), desplegará debajo un menú de búsqueda. En este caso busque la palabra copy, la podemos ver resaltada en el listado.

#### **3.1.5.** ¿Cómo puedo buscar la siguiente palabra clave, buscada anteriormente?

Para seguir avanzando en todas las coincidencias de la palabra buscada debemos simplemente tipear”s” y darle a enter.

#### **3.1.6.** ¿Cómo puedo salir de Info? -hágalo.

Para salir de info se debe utilizar la letra Q.

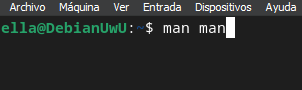
### **3.2.** MAN: man es un programa que formatea y muestra las páginas del manual.

#### **3.2.1.** ¿Cuál es la diferencia entre man e info?

|  |  |
| --- | --- |
| MAN (Manual Pages/Páginas de manual) | INFO (information/información) |
| Proporciona páginas de manual que contienen información sobre comandos, llamadas del sistema, bibliotecas, archivos de configuración y otros aspectos del sistema. | Proporciona documentación más extensa y detallada, a menudo en forma de manuales de usuario o guías de referencia que explican el uso de programas, herramientas y funciones. |
| Utiliza un formato de texto plano con secciones bien definidas como NAME, SYNOPSIS, DESCRIPTION, OPTIONS, y EXAMPLES. | Utiliza un formato estructurado en nodos organizados en una estructura de árbol, con hipervínculos entre nodos. |
| Las páginas de manual están estructuradas en secciones y se navega típicamente usando comandos como space (para avanzar), b (para retroceder), y q (para salir). | Permite una navegación más dinámica con comandos para moverse entre nodos, buscar términos específicos y navegar a través de enlaces. |

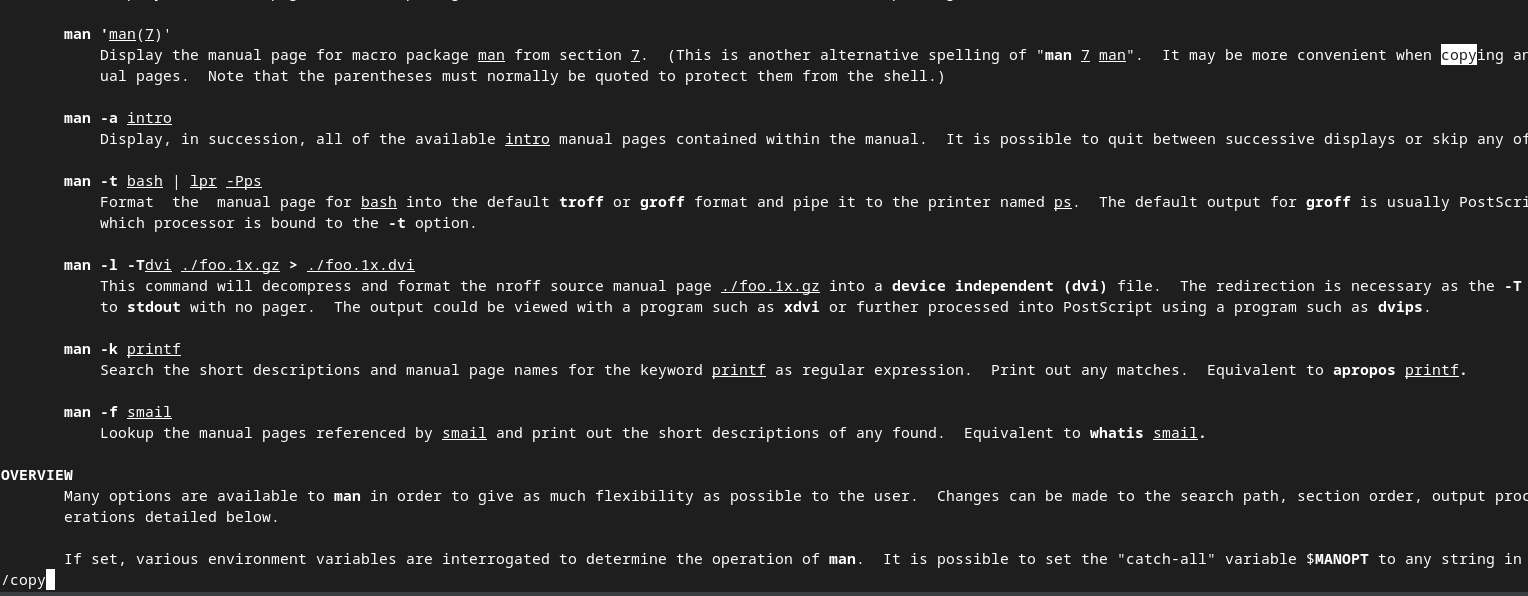
#### **3.2.2.** ¿Cómo puedo ver la información de un determinado comando?

Para ver información de un comando en específico debemos escribir el comando “man” seguido del comando que deseamos buscar, en este caso, visualizamos información sobre man al escribir “man man”



#### 3.2.3. ¿Cómo puedo buscar una palabra clave dentro de la página del manual?

Para buscar una palabra clave en la página debemos utilizar la barra / seguida de la palabra clave. En este caso utilice “copy” como palabra clave, podemos verla remarcada.

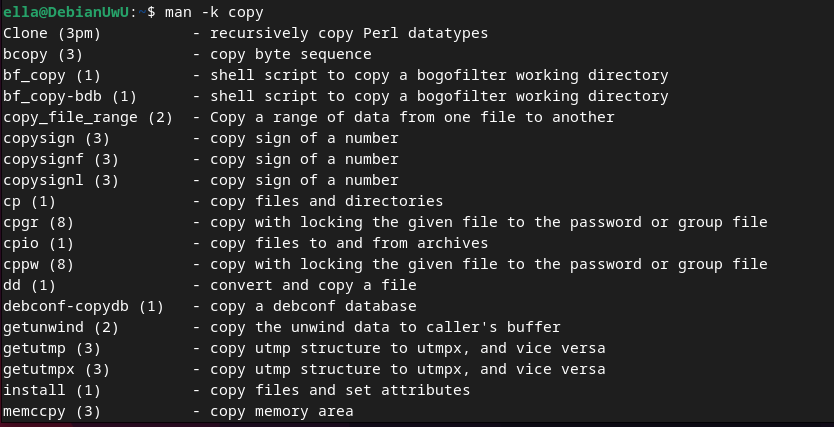


#### 3.2.4. ¿Cómo puedo salir?

Salimos mediante la letra Q.

#### 3.2.5. ¿Cómo hago para buscar una palabra clave determinada en todas las páginas del manual?

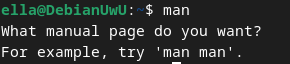
Podemos buscar una palabra en todas las páginas mediante el comando “man -k”, seguido de la palabra que deseamos buscar. En este caso utilice copy.



#### 3.2.6. ¿Qué es lo sucede al realizar lo siguiente?

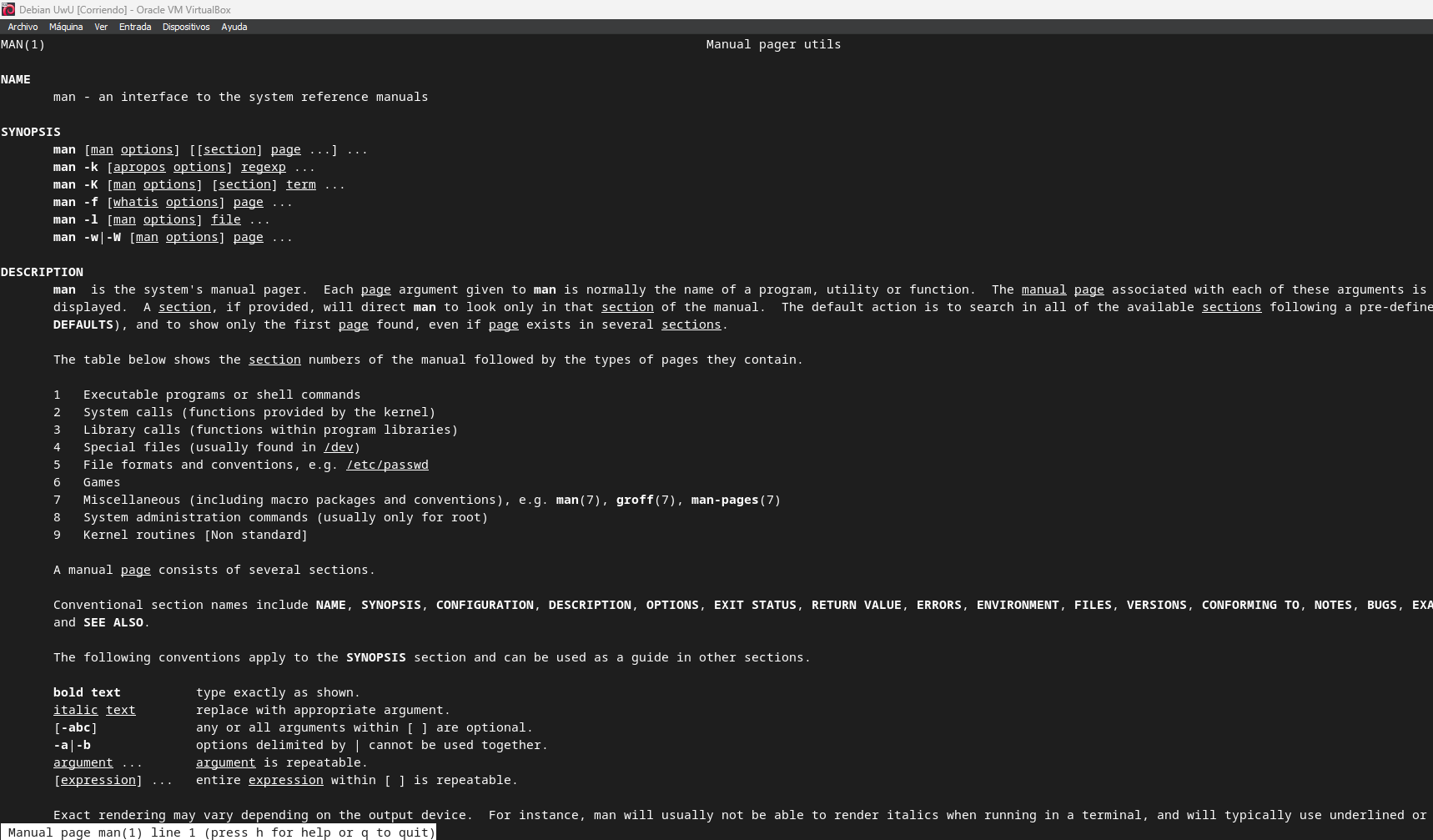
##### 3.2.6.1. man

Se nos consultara que pagina deseamos consultar mediante man.



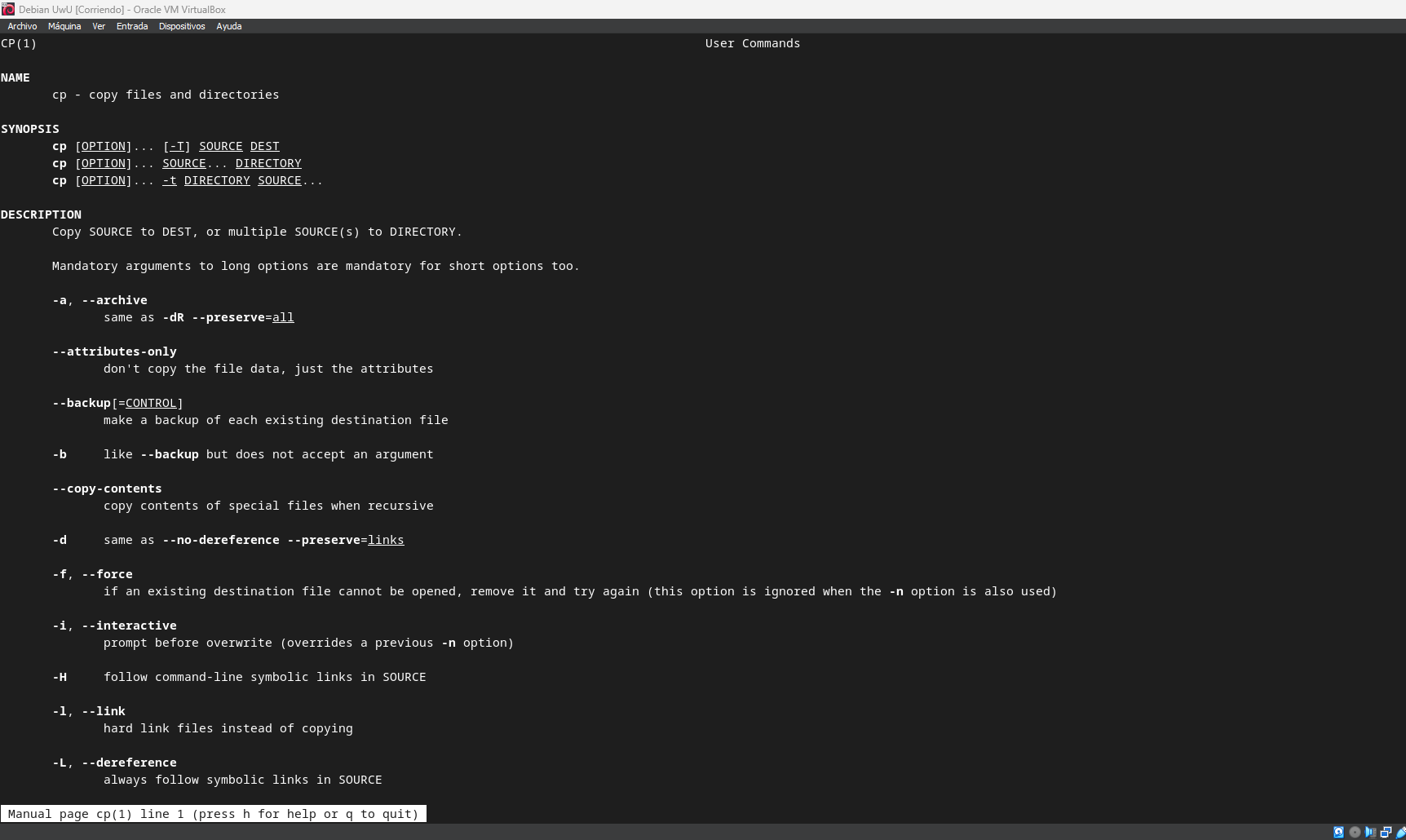
##### 3.2.6.2. man man

Se consultará la pagina de man.

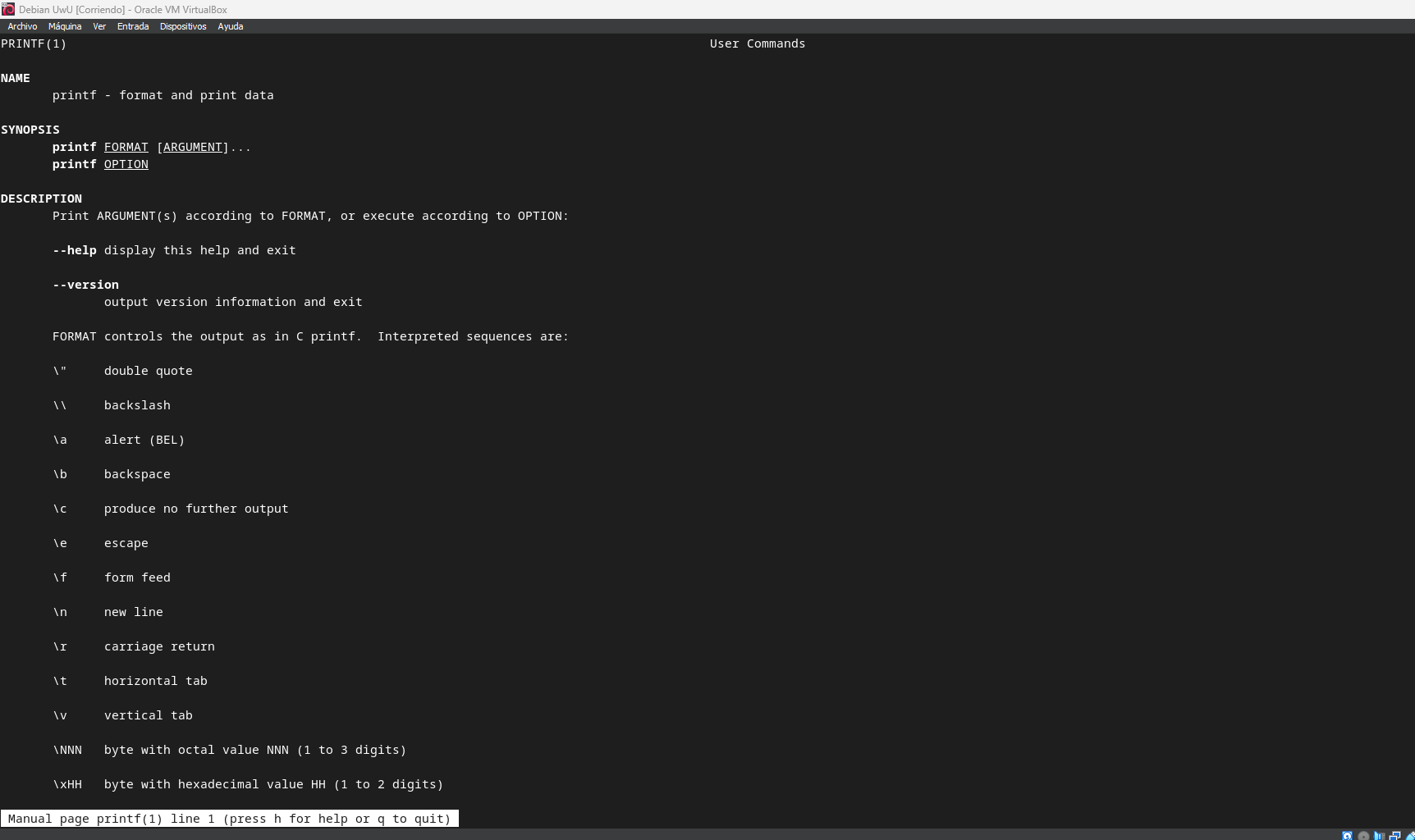


##### 3.2.6.3. man cp

Se consultara la pagina del comando cp.



##### **3.2.6.4.** man printf



##### **3.2.6.5.** man fprintf



##### **3.2.6.6.** man sprintf



##### **3.2.6.7.** man 3 printf



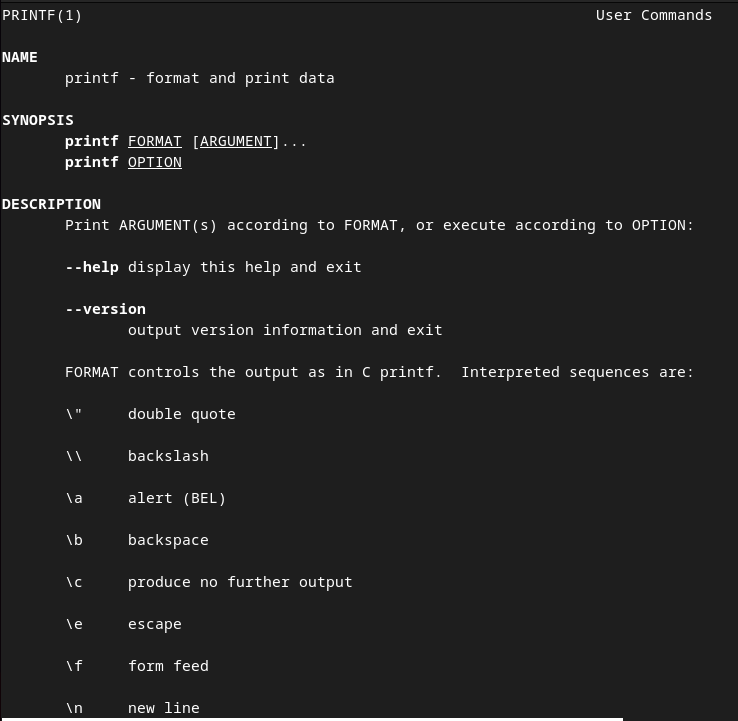
#### **3.1.2.** Del punto anterior, responder:

##### **3.1.2.1.** Al invocar man junto con fprintf y sprintf muestra la misma página. ¿Porqué no muestra la misma página al invocarlo con printf?. (TIP: vea el punto 3.2.6.2).

fprintf y sprintf comparten pagina del manual, al igual que lo hacen con printf, sin embargo la pagina compartida corresponde a la Libreria de Funciones en C. Podemos apreciar que no se da una explicación detallada de los distintos tipos de “printf”, si no que se hace una explicación detallada de su implementación en C.

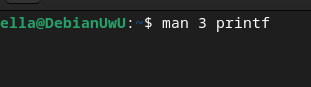


La página única de printf esta catalogada como comandos de usuario, por lo tanto, se documenta a printf de una forma mas detallada, haciendo hincapié en su funcionalidad para imprimir información y no su uso en dentro de la librería correspondiente a C.



##### **3.1.2.2**. ¿Cómo puedo invocar al man para ver directamente la función printf del lenguaje C?.

Lo podemos hacer anteponiendo un 3 a la funcion de printf, especificando que deseamos ver la sección 3 del manual.



### **3.2.** HELP: help es la ayuda que ofrece el shell de GNU/LINUX para utilizar sus comandos.

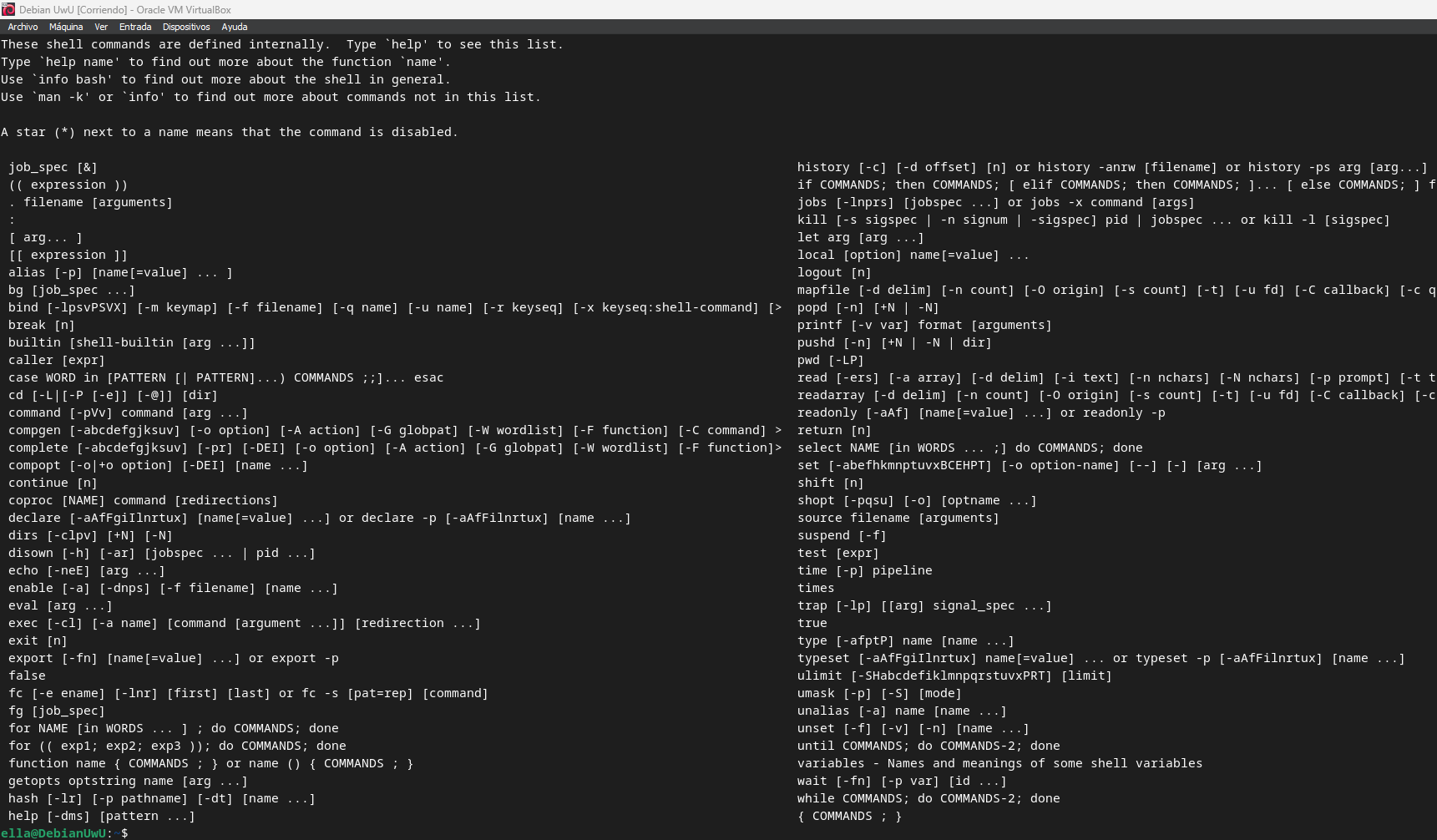
#### **3.2.1.** ¿Cuál es la diferencia entre help e info?.

|  |  |
| --- | --- |
| HELP (ayuda) | INFO (información) |
| El comando help se utiliza para obtener información rápida sobre los comandos integrados de la shell de Unix/Linux (comandos internos del shell). | El comando info se utiliza para acceder a la documentación más detallada y extensiva proporcionada por el sistema GNU. La documentación en el formato Info puede ser más extensa y estructurada, proporcionando una vista completa del uso y las opciones de los programas. |
| La información proporcionada por help suele ser breve y está destinada a proporcionar una descripción rápida del comando y sus opciones. | La documentación de info está organizada en una estructura de menús y submenús, lo que permite una navegación más extensa a través de la información. |
| Está disponible en shells como Bash y Zsh. | Está disponible en sistemas que usan la documentación de GNU, como la mayoría de distribuciones de Linux. |

#### **3.2.2.** ¿Cuál es la diferencial entre help y man?.

|  |  |
| --- | --- |
| HELP (ayuda) | MAN (manual) |
| Proporciona información sobre los comandos internos de la shell (como cd, echo, etc.) y algunas características básicas de la shell misma. | Proporciona documentación detallada sobre una amplia gama de comandos y programas disponibles en el sistema, incluidos tanto comandos internos del sistema como programas externos. |
| Está limitado a los comandos incorporados en la shell y no cubre comandos externos ni programas de usuario. | Cubre comandos externos, funciones de bibliotecas, configuraciones del sistema y otros aspectos detallados de programas y herramientas. |
| La información suele ser breve y directa, enfocada en la sintaxis y opciones básicas del comando. | La documentación está organizada en páginas de manual divididas en secciones, cada una con información extensa sobre el comando, sus opciones, ejemplos y más. |

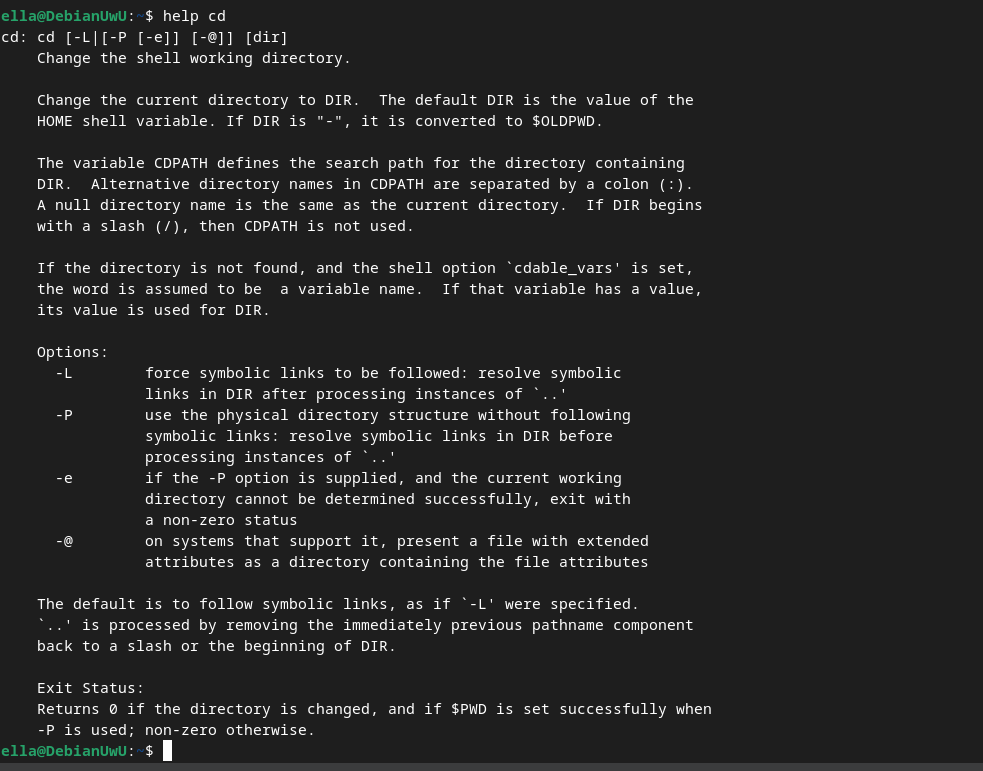
#### **3.2.3.** ¿Qué sucede al invocar al help?.



#### **3.2.4.** ¿Cómo puedo ver la información de un determinado comando?

Se debe tipear help seguido del comando deseado.

#### **3.2.5.** ¿Qué sucede al utilizar el comando help cd?



### **3.3.** whereis

#### **3.3.1.** ¿Qué sucede al utilizar el comando whereis cp?



#### **3.3.2.** ¿Qué es la información que se muestra por pantalla al ejecutar el punto anterior?

**/bin**: Es un directorio que contiene binarios esenciales que son necesarios para el sistema operativo, especialmente para el arranque y la reparación del sistema. Los binarios en este directorio son accesibles a todos los usuarios del sistema.

**cp**: Es el archivo ejecutable para el comando cp. Este comando se usa para copiar archivos y directorios en sistemas Unix/Linux.

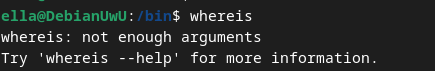
**/usr/share**: Es un directorio que contiene archivos de datos que son compartidos entre aplicaciones. Este directorio suele incluir documentación, imágenes y otros archivos de recursos.

**man**: Es un subdirectorio que contiene las páginas de manual del sistema. Las páginas de manual se utilizan para proporcionar documentación sobre comandos y programas.

**man1**: Es un subdirectorio dentro de /usr/share/man que contiene las páginas de manual para comandos de usuario, específicamente en la sección 1 del manual. La sección 1 generalmente cubre comandos de usuario.

**cp.1.gz**: Es el archivo comprimido de la página de manual para el comando cp, en la sección 1 del manual. La extensión .gz indica que el archivo está comprimido en formato gzip para ahorrar espacio.

#### **3.3.3.** ¿Qué ocurre si se ejecuta whereis \* sobre un directorio? (**Tip**: si no pasa nada, intentalo nuevamente pero primero ejecute cd /bin)

****

#### **3.3.4**. ¿Cuál es la diferencia entre whereis y find?

**whereis:** Busca ubicaciones específicas de archivos relacionados con comandos, es decir, es ideal para encontrar ubicaciones de binarios y archivos de documentación.

**find:** Proporciona una búsqueda exhaustiva y flexible en el sistema de archivos, permitiendo buscar basándose en varios criterios y acciones adicionales.

### ***3.4.*** whatis

#### *3.4.1.* ¿Qué sucede al utilizar el comando whatis cp?

What Is dará una breve descripción de que hace el comando CP

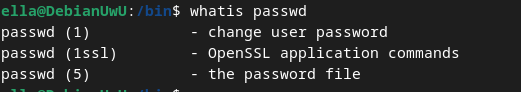


#### ***3.4.2.*** Cambie al directorio /bin, y ejecute el comando whatis \* ¿Qué ocurrió?



#### ***3.4.3.*** Utilice el comando apropos passwd y whatis passwd. Enumere las diferencias encontradas en el resultado de cada uno de los comandos.





**apropos:** Busca en todas las descripciones de páginas de manual para encontrar coincidencias que contengan el término de búsqueda, esto también puede incluir secciones, por lo que mostrará múltiples resultados.

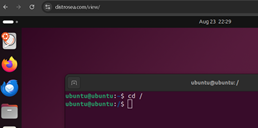
**whatis:** Busca una entrada exacta en la base de datos de manuales para el término especificado, por lo que se mostrará una sola coincidencia con una descripcion breve.

## **4. DIRECTORIOS**

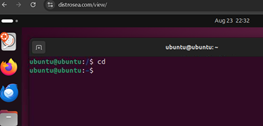
### **4.1.** ¿Para qué se usa el comando cd? Ejecute las siguientes variantes de cd e indique cuál fue el resultado obtenido:



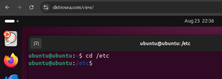
#### **4.1.1.** cd /



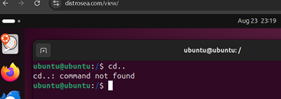
#### **4.1.2.** cd



#### **4.1.3**. cd /etc



#### **4.1.4.** cd..



#### **4.1.5.** cd ..



### **4.2.** Bash sobre directorios:

#### **4.2.1.** ¿Cuáles son las diferencias entre el path absoluto y el path relativo?

Path significa “ruta”.

Un Path se compone de un nombre y la barra diagonal. Como por ejemplo */home/indu/scripts*

Cuando comienza con un “/” significa que es la raíz por ejemplo /home/indu/scripts

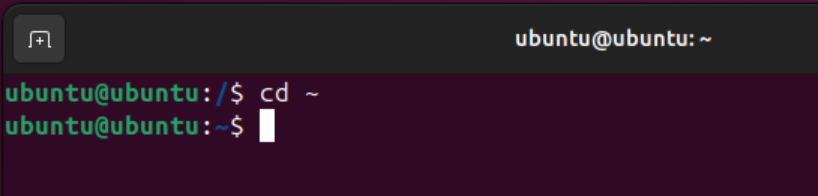
**Path Absoluto**

Siempre comienza desde el directorio raíz (/).Por ejemplo, /home/indu/scripts/mis\_scripts.sh.

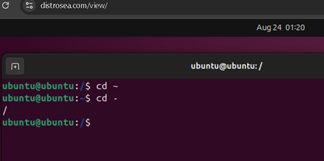
**Path relativo**

Comienza desde el directorio actual donde estoy trabajando. Por ejemplo, si estoy en la carpeta indu y quiero acceder al archivo mis\_scripts.sh.debería escribir/scripts/mis\_scripts.sh.

##### **4.2.2.1.** cd ~



##### **4.2.2.2**-cd –



#### **4.2.3** **¿Cuál es la diferencia entre cd ~ y cd -?**

**cd ~**

Este comando te lleva al directorio de inicio (home) del usuario actual. El símbolo ~ es un atajo que representa el directorio home.

Por ejemplo, si tu nombre de usuario es miguel, ejecutar cd ~ te llevará a /home/miguel.

**cd -**

Este comando te lleva al último directorio en el que estabas antes del directorio actual. De esta manera es útil para alternar rápidamente entre dos directorios.

Si estás en /home/miguel/docs y te mueves a /var/log con cd /var/log, al ejecutar cd - regresarás a /home/miguel/docs. Si vuelves a ejecutar cd -, regresarás a /var/log.

### **4.3** Operaciones con directorios:

#### **4.3.1** ¿Con que comando se puede **crear** un directorio?

El comando para **crear** Carpetas o Directorios esmkdir.

A saber, “mkdir” traduce a “make directory”.

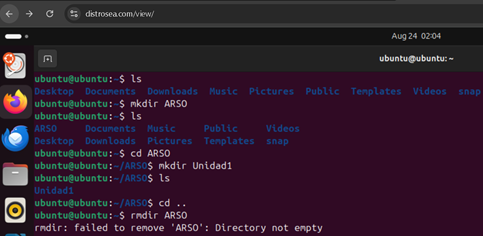
#### **4.3.2** ¿Con qué comando se puede **borra**r un directorio?

El comando para poder **eliminar** Directorios o Carpetas es rmdir.

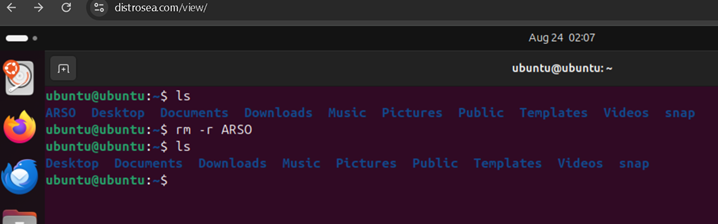
A saber, “rmdir” traduce a “remove directory”

#### **4.3.3**¿Qué sucede sí el directorio no está vacío?

Si el directorio no está vacío el comando rmdir no va a funcionar. Ya que puede borrar directorios vacíos.

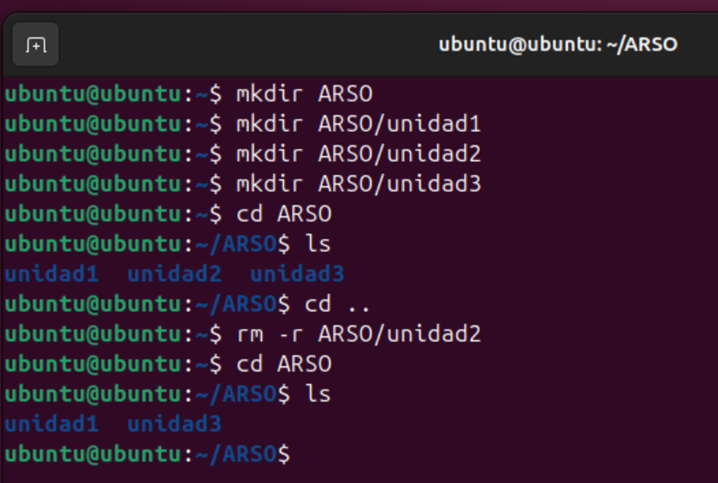


Para borrar un directorio que contiene archivos o subdirectorios, debes usar el comando rm con la opción -r (que traduce a “recursive”), de la siguiente manera:



#### **4.3.4**¿Cómo puedo salvar la situación anterior? (Sin borrar uno a uno los archivos existentes).

Este comando protege contra el borrado de múltiples archivos dentro de un directorio específico. La sintaxis es rm –r DIRECTORIO/carpeta\_que\_quiero\_borrar



### **4.4.** ¿Qué significa la expresión ./ cuando se utiliza delante de un archivo? ¿Para que sirve?

La expresión ./ indica el directorio en donde estoy actualmente.

Representemos la situación con un ejemplo.

Puedo crear una carpeta llamada “ARSO” en la cual habrá 3 carpetas más “unidad1”, “unidad2”, “unidad3”. Dentro de “unidad”3 habrá un script llamado “mi\_script.sh”.

Podría estar ahora en mi carpeta “ARSO” y utilizar ./ para indicar que estoy sobre esa ruta de esta manera: ./unidad3/mi\_script.sh

### **4.5.** ¿Cómo puede moverse entre directorios sin utilizar el PATH completo?

Puedes moverte entre directorios sin utilizar la ruta completa (PATH) usando el comando cd. Algunos ejemplos:

* Para moverte un nivel hacia arriba utilizando cd ...
* Para moverte a un subdirectorio dentro del directorio actual, solo escribe cd nombre\_del\_directorio.

Por ejemplo los comando del punto 4.2.3 son utilizados para:

* Para ir directamente a tu directorio personal, usa cd ~
* Para volver al directorio anterior utiliza cd -.

Estos comandos te permiten navegar de manera eficiente sin necesidad de escribir la ruta completa.

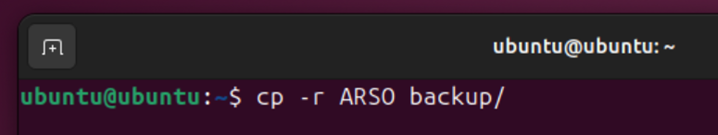
## **5. ARCHIVOS**

### **5.1**¿Qué hacen los siguientes comandos?

#### **5.1.1.** Cp

Cp traduce a copy y permite realizar copias de carpetas enteras o de archivos individuales.

Por ejemplo siguiendo el ejemplo del punto 4.4. Si deseo copiar la carpeta ARSO con sus subdirectorios a otra llamada “backup” tengo que usarlo de esta manera.



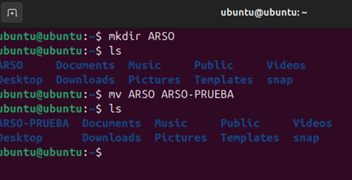
La expresión -r traduce a “recursivo” y es usada para copiar directorios enteros.

Si quiero copiar archivos individuales puede utilizar cp.

#### **5.1.2.** **mv**

Mv traduce a move usándose para dos funciones.

La primera es renombrar un archivo o un directorio de esta manera,



Otro uso que se le puede dar es el de mover un archivo o un directorio especificando la ruta por ejemplo: mv carpeta\_origen/ /rutal/al/destino

#### **5.1.3**.rm

Rm traduce remove

Este comando rm borra el fichero con el nombre especificado. Pero el comando rm –r borra el directorios especificado con todo su contenido.

#### **5.1.4** rcp

Rcp traduce a remote copy

Este comando se utiliza para copiar archivos desde un sistema a otro.

Éste utiliza los archivos /etc/hosts. equiv y /etc/passwd del sistema remoto para determinar si dispone de derechos de acceso al sistema remoto.

Veamos las dos situaciones posibles

1.Copiar un archivo desde tu sistema local a un sistema remoto llamado “servidor2” en un directorio llamado “var/tmp”. rcp /home/usuario/archivo\_local.txt servidor2:/var/tmp/

2. Copiar un archivo desde un sistema remoto a tu sistema local. El archivo que deseo copiar se llama “archivo\_remoto.txt” y está en mi sistema remoto llamado “servidor1”. Este archivo quiero copiarlo en el directorio “/home/usuario/”

rcp servidor1:/ruta/del/archivo/archivo\_remoto.txt /home/usuario/

#### **5.1.5**. rsh

Rsh traduce a remote shell

La herramienta rsh se utilizaba para ejecutar un único comando en un Sistema. Fue introducida en la comunidad en BSD Unix en la década de los 80.

Rsh usaba archivos .rhost y etc/hosts.equiv para la autenticación. Estos métodos se basan en direcciones IP y DNS para autenticase.  
 El punto en contra de esta herramienta es que la falsificación (spoonfing) de la dirección IP es sencilla para personas experimentadas.

Posteriormente fue reemplazada por ssh siendo esta mas segura.

#### **5.1.6** scp

scp traduce a source copy

Se utiliza para copiar archivos y directorios entre sistemas, pudiendo ser locales o remotos. Utiliza el protocolo SSH (Secure Shell) para garantizar que la transferencia de datos esté encriptada y sea segura.

1-Desde sistema local a uno remoto: scp archivo usuario@remoto:/ruta/destino

2-Desde un sistema remoto a uno local: scp usuario@remoto:/ruta/archivo /ruta/local/destino

Además, nos ofrece la posibilidad de interactuar entre dos sistemas remotos.

#### **5.1.7.** **ssh**

Ssh traduce a Secure Shell

Es un protocolo utilizado para, de manera segura, accede desde una computadora a otra.

Nos permite utilizar líneas de comandos, transferir y proporcionar comunicaciones cifradas seguras entre dos hosts que no tienen seguridad.

Con el cifrado que proporciona, garantiza que tus datos estén protegidos mientras se transmiten por la red, incluso en entornos no seguros.

El usuario debe validar su identidad a la máquina remota utilizando algún protocolo de autenticación.

Es un reemplazo de rlogin, rsh y rcp

### **5.2.** ¿Con qué comando puedo concatenar el contenido de dos archivos?

Si tengo 2 archivos uno es llamado “archivo1.txt” y el otro “archivo2.txt” y mostrarlo usaría el comando cat de esta manera:

cat archivo1.txt archivo2.txt

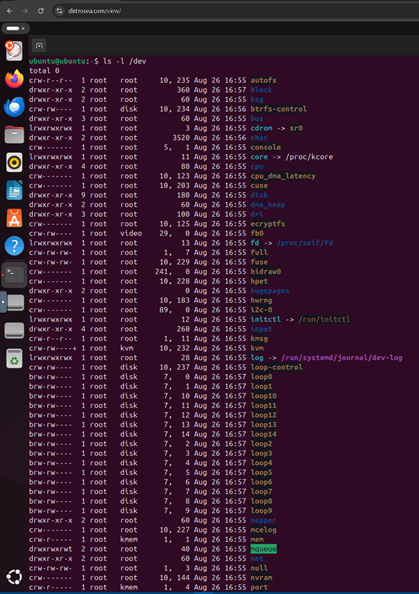
Si deseo concatenar pero hacer un nuevo archivo llamado “archivo\_combinado.txt” que posea todo el contenido de ambos archivos, el comando cat seria de esta manera:

cat archivo1.txt archivo2.txt > archivo\_combinado.txt

#### **5.2.1**. ¿Se puede usar ese comando para otra cosa?

Si, puedo utilizarlo para realizar una concatenación múltiple de archivos como también, lo ya mencionado, transferir la salida a otro archivo nuevo.

### **5.3** Haga un ls –l /dev



#### **5.3.1** ¿Qué significa el primer carácter?

El primer carácter pertenece a la cadena de permisos e indica el tipo de archivo que es.

#### **5.3.2.** ¿Cuáles son todos los posibles valores que puede contener ese campo y que

¿Qué significa cada uno?

· “-“: significa que es un archivo regular.

· “d”: hace referencia a un directorio.

· “l”: representa a un enlace simbólico. Haciendo referencia a otro archivo.

· “s”: refiere a socket

· “c”: hace referencia a un archivo especial de caracteres.

· “b”: referencia a un dispositivo especial de bloque.

· “p”: canal, referenciando a un archivo FIFO o tubería “pipe”

### **5.4 ¿Para qué sirve el comando touch? ¿qué utilidad le encuentra?**

El comando touch en Linux tiene dos usos principales:

1. **Actualizar la fecha y hora de acceso y modificación de un archivo existente:** Si el archivo ya existe, touch actualiza la fecha y hora de acceso y modificación a la hora actual. Esto es útil, por ejemplo, si quieres que un archivo parezca recientemente modificado sin cambiar su contenido.

Ya vimos que mkdir era para crear carpetas. En este caso si ya existiese un archivo llamado “archivo.txt” utilizando touch archivo.txt actualizará la hora y fecha del archivo a la actual.

1. **Crear un archivo vacío si el archivo no existe:** Si el archivo no existe, touch creará un archivo vacío con el nombre especificado. Esto es útil para generar rápidamente archivos nuevos.

Ya vimos que mkdir era para crear carpetas. En este caso si no existe el archivo, podríamos crear uno con el comando: touch nuevo\_archivo.txt

## **6.PERMISOS**

### **6.1.** Teniendo en cuenta el ls –l anterior, ¿indique que son los siguientes 9 caracteres? (sin considerar el primero sobre el que ya respondió anteriormente)

El primer grupo de tres (rwx)nos dice que clase de permisos tiene el dueño del fichero (u) (user/owner)

El segundo grupo de tres ( r-x) nos dice que clase de permisos tiene el dueño del fichero (g) (group).

Y el ultimo grupo de tres(--- ) nos dice que clase de permisos tienen todos los demás usuarios del sistema sobre este fichero (o) (others).

### **6.2.** ¿qué significan cada caracter? ¿cómo están agrupados?

Estan agrupados en grupos de a 3:

r: significa permiso para leer.

w: significa permiso para escribir.

x: significa permiso para ejecutar.

### **6.3.** ¿Cómo se asignan los permisos? (detalle los comandos).

Chmod [opciones] modo-en-octal fichero

Las opciones podemos identificarlas o no, según queramos. Opciones típicas son:

-R para que mire también en los subdirectorios de la ruta.

-v para que muestre cada fichero procesado.

-c es como –v, pero solo avisa de los ficheros que modifica sus permisos.

El modo en octal es un numero en base 8 (octal) que especifique el permiso. Los números en octal se especifican empezando el numero por un 0 . Por ejemplo,0777 es indica, todos los permisos posibles para todos los tipos de usuario. 0666 indica que se dan permisos de lectura y escritura, pero no de ejecución. 0766 indica que se dan permisos de lectura y escritura, pero solo tienen permiso de ejecución para los usuarios que son dueños del archivo. 0755 indica permisos para lectura y ejecución, pero escritura solo para el usuario que es dueño del archivo.

Por ejemplo

Chmod 0777 archivo.txt

Asigna todos los permisos al archivo archivo.txt

Chmod 0666 \*

Asigna permisos de lectura y escritura, no de ejecución a todos los archivos y directorios del directorio donde ejecutamos el comando.

Chmod -R 0644 \*

Esto da permisos a todos los archivos y directorios del directorio donde se invoca el comando y de todos los directorios que cuelgan de el. Los permisos asignados son de lectura a todos los usuarios, de escritura solo al dueño del archivo y de ejecución a nadie.

Otro modo de trabajo con chmod

Luego tambien se pueden asignar permisos de otra manera, utilizando otra posible sintaxis de chmod, que tal vez resulte mas util si no queremos tratar con los valores en octal.

chmod [opciones] modo [,modo]…fichero

Para ello tenemos que tener claro los distintos tipos de usuario:

u: usuario dueño del fichero.

g: grupo de usuarios del dueño del fichero.

o: todos los otros usuarios.

a: todos los tipos de usuarios(dueño ,grupo y otros)

También hay que saber la letra que abrevia cada tipo de permiso:

r: se refiere a los permisos de lectura.

w: se refiere a los permisos de escritura.

x: se refiere a los permisos de ejecución.

Ejemplos

chmod o=rwx \*

Asigna permisos de lectura ,escritura y ejecución para los usuarios “otros” a todos los archivos de la carpeta.

chmod a=rwx fichero.txt

Asigna todos los permisos a todos los usuarios para el archivo fichero.txt

chmod go= \*

Quita todos los permisos para los usuario del grupo y los usuarios otros.

chmod u=rwx,g=rw,o= \*

Da todos los permisos al dueño del fichero, a los del grupo del dueño le asigna permisos de lectura y escritura y a los otros usuarios les quita todos los permisos.

Nota:un espacio después de la coma "," en los distintos modos de permisos que se indiquen hace fallar el comando.

chmod a=r \*

Da permisos únicamente de lectura a todos los tipos de usuario.

De un modo parecido a lo que acabamos de ver, también se pueden añadir o quitar permisos con los operadores + y -. Para ello se indica el tipo de usuario y el permiso que se resta o añade. Algo como esto:

chmod a-wrx \*

Esto quita todos los permisos a todos los tipos de usuario.

chmod a+r,gu+w \*

Este comando asigna permisos de lectura a todos los usuarios y permisos de escritura al dueño del archivo y el grupo del dueño.

chmod u=w,a+r \*

Este comando asigna permisos de escritura al usuario dueño y a todos los usuarios les añade permiso de lectura.

### **6.4.** ¿Qué son el owner, y el group de un archivo? ¿Se pueden cambiar?

Owner es el propietario de un archivo.

Group es el grupo al que pertenece un archivo.

Si se pueden cambiar. Para cambiar el dueño del fichero se utiliza el comando : chown usuario fichero.

Para cambiar el grupo del fichero se utiliza el comando: chgrp grupo fichero

Para cambiar los permisos se utiliza el comando: chmod permisos fichero.

### **6.5.** Intente cambiar los permisos de un archivo perteneciente al root (sesionado como usuario). Explique qué sucedió.

root@luz-desktop:/# ls -l /home/luz/Escritorio

total 0

-rw-r--r-- 1 luz luz 0 2024-08-30 19:30 lala

root@luz-desktop:/# cd home

root@luz-desktop:/home# cd matias

root@luz-desktop:~# cd Escritorio

root@luz-desktop:~/Escritorio# sudo chmod ugo+rxw lala

root@luz-desktop:~/Escritorio# ls -l

total 0

-rwxrwxrwx 1 luz luz 0 2024-08-30 19:30 lala

### **6.6.** Explique la forma de cambiar los permisos con valores en octal.

chmod 750 lala

da permisos rwx al usuario (7=111)

da permisos r-x al grupo (5=101)

da permisos --- a los demás (0=000)

### **6.7.** ¿Cuál es el significado de los permisos en los directorios (se debe indicar que indica una r, una w, y una x)?

R= Permiso para leer el archivo

W= Permiso para escribir el archivo

X= Permiso para ejecutar el archivo

## **7.FILTROS**

### **7.1.**¿Cuál es la diferencia de los comandos more, less y cat? De un ejemplo de cada uno.

El comando less sirve para mostrar el contenido de un archivo o directorio de a poco.

El comando more sirve para mostrar el contenido de un archivo o directorio de a tandas teniendo que presionar enter para seguir viendo el contenido, e informa cuanto contenido del total se ve.

Y el comando cat sirve para mostrar el contenido de un archivo o directorio en forma total sin tener que presionar enter como lo hacia con more.

### **7.2.**¿Cuál es la diferencia entre tail y head?

Head muestra la primera parte del archivo.

Y Tail muestra la parte final del archivo.

### **7.3.**¿Qué es lo que realiza el comando uniq?

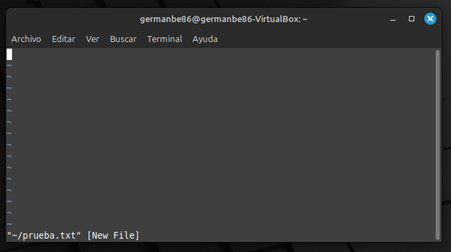
Uniq omite las lineas que estén repetidas en un archivo, es decir no las muestra.

### **7.4.**¿Qué es lo que realiza el comando grep?

El comando Grep muestra las lineas de un archivo en las cuales se comparta una palabra en comun.

## 8. VI

### **8.1.** Ejecute la siguiente instrucción: vi $HOME/prueba.txt ¿Qué sucedió? Ahora ejecute todos los pasos detallados a continuación.

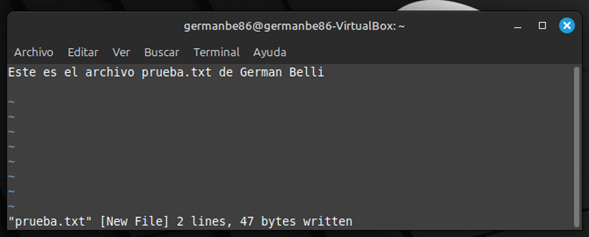


Se creó un archivo en el directorio Home llamado prueba.txt y se abrió con el editor de Texto de la consola de Linux llamado “VI”.

### **8.1.1.** Escriba la siguiente frase: “Este es el archivo prueba.txt de <nombre y apellido>”

### **8.1.2.** ¿Qué tuvo que hacer para poder escribir la frase?

Tuve que ingresar en el modo de entrada de texto (tocando la tecla “i”) y luego pude escribir.



### **8.1.3.** Guarde el archivo, y salga del editor. ¿Qué comando utilizó?

Primero toque ESC para salir del modo de entrada y volver a modo comando. Luego “:w” para guardar y luego salí con :”q”

### **8.1.4.** Ingrese nuevamente al archivo.

Lo hago mediante el comando: “vi prueba.txt “

### **8.1.5.** Incorpore al inicio del archivo el siguiente párrafo (los acentos pueden ser evitados):

“Sistemas Operativos

Curso de los días <día de cursada>

Trabajo Práctico 1

Alumno: <su nombre y apellido aquí>

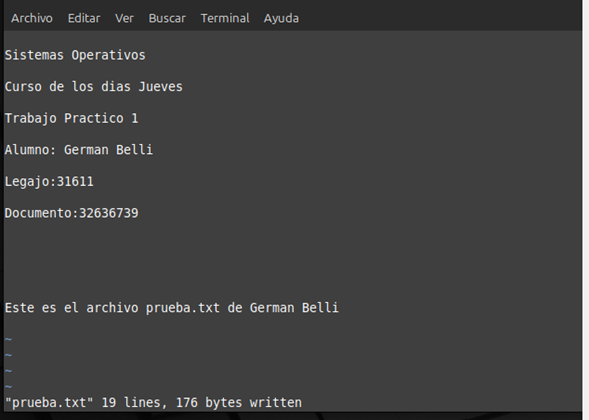
Legajo: <su Legajo aquí>

Documento: <su documento aquí>”

### **8.1.6.** Describa todos los pasos que tuvo que realizar.

- Escribo por consola: “vi Prueba.txt “ (siempre parado en el directorio donde se encuentra dicho archivo)

- paso a modo de entrada con la letra ”i” .Me posiciono delante del texto antes creado y le doy ENTER para que baje unos renglones.

- Comienzo a escribir lo que dice el ejercicio- Si quiero borrar una línea, en modo comando pulso “dd”

### **8.1.7.** Guarde el archivo y continúe la edición. ¿Qué comandos utilizó?

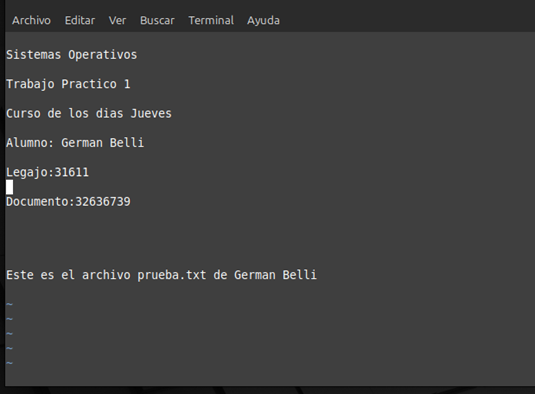
- Al finalizar toco “:w” para guardar y continuar con la edición. “:q”para salir. Tambien se puede usar el comando “:wq” si desea guardar y salir directamente.

### **8.1.8.** Borre la línea de “Legajo”. Indique por lo menos dos formas de realizarlo.

- Me paro en esa línea en modo comando y pulso “dd” o “D”

-Tambien podría posicionarme en ese renglón en modo entrada,adelante del texto, e ir borrando con el botón Suprimir, carácter por carácter.

### **8.1.9.** Invierta el orden de las líneas “Curso y TP”. No está permitido rescribirlas. ¿Qué comandos utilizó?



Debemos colocarnos en la primera línea que contiene el texto que nos interesa y pulsar:

1º el número de líneas a utilizar y si queremos copiarlas (*yy*) o cortarlas (*dd*) (Ejemplo pulsar “2yy” para copiar 2 lineas)

**2º p** si queremos pegar DEBAJO de la línea actual.

**3º P** para pegar ENCIMA de la línea en la que nos encontramos.

### **8.1.10.** Ubíquese en la línea 2 (dos) del archivo. No está permitido usar las teclas del cursor, ni el mouse. ¿Qué comando utilizó?

- Primero el número “2” y luego “shift+g”

### **8.1.11.** Marque para copiar las líneas 2, 3, y 4 (todas juntas, no de a una a la vez). ¿Cómo lo realizó?

Me posiciono con el cursor en la línea 2 y presiono “3yy”

### **8.1.12.** Ubíquese al final del archivo (sin usar las teclas del cursor), y pegue dos veces el contenido del buffer. ¿Qué comando usó?

**Primero presiono “G” y luego”pp”**

### **8.1.13.** Deshaga uno de los copiados. No está permitido borrar línea por línea, ni carácter a carácter. ¿Qué comando usó?

**Solo presionando “u”**

### **8.1.14.** ¿Cómo busco la palabra “Documento”? ¿Cómo busco la segunda ocurrencia de una palabra?

Para encontrar un string, hay que escribir una barra hacia la derecha (/) seguida por el string que se desea buscar, y pulsar return. El vi posiciona el cursor en la siguiente ocurrencia del string. Tecleando n se va a la siguiente ocurrencia del string, y pulsando N a la anterior.

### **8.1.15.** ¿Cómo puedo reemplazar la palabra “Documento” por “Documento:” (sin borrar, o realizar el reemplazo a mano)’

Para reemplazar una palabra, sitúe el cursor al principio de la palabra que va a ser reemplazada. Escriba cw, seguido de la palabra nueva. Para terminar, presione Escape.

### **8.1.16.** Guarde el archivo y salga.

“:wq”

### **8.1.17.** Ejecutar "vi miprog.c"

introducir el texto:

principal ()

{

printf (" 1/3/2001 ------------- listo");

printf (" 1/3/2002 ------------- listo");

printf (" 1/3/2003 ------------- listo");

printf (" 1/3/2004 ------------- listo");

printf (" 1/3/2005 ------------- listo");

printf (" 1/3/2006 ------------- No listo");

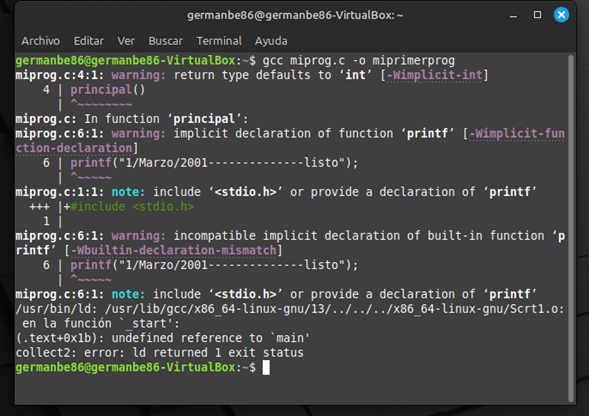
}

Luego presionar “:x”

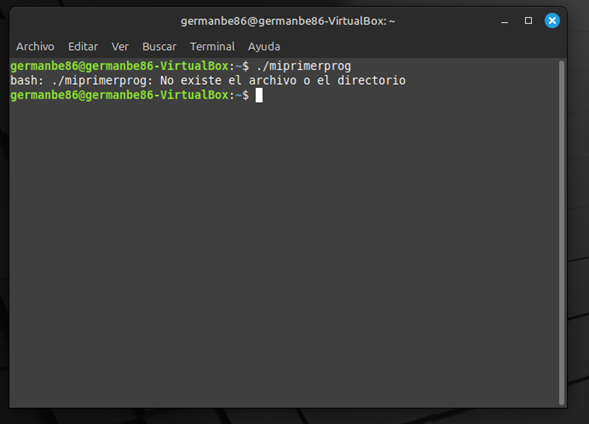
Ahora ejecute los siguientes comandos:

“gcc miprog.c -o miprimerprog” (si no tiene instalado gcc, hágalo usando el comando sudo apt install gcc). GCC es el compilador de Linux.

“./miprimerprog”

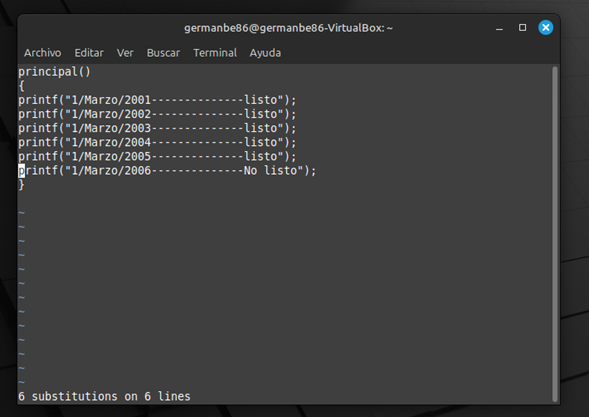
¿Cuál es la salida? “./miprimerprog/”

Como no pudo compilar, no puede correr un ejecutable que todavía no existe.

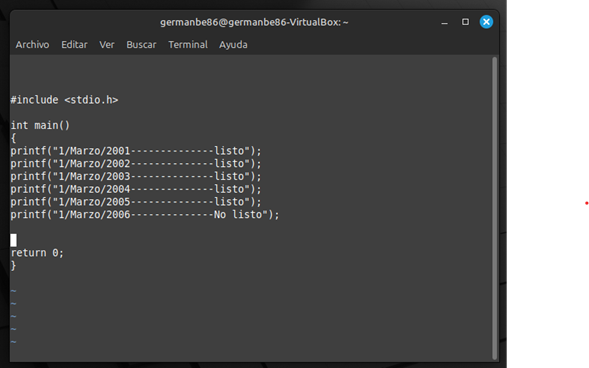


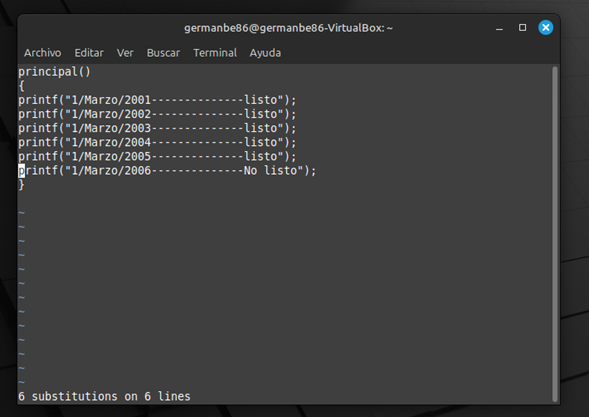
volver a ejecutar: “ vi miprog.c”

poner “:%s/\/3/\/Marzo/g” ¿Que paso al ejecutar esto?

Se sustituyó el mes en números, por el nombre del mes en cada renglón de la planilla.

Corrija los errores que surgieron al utilizar el comando gcc.



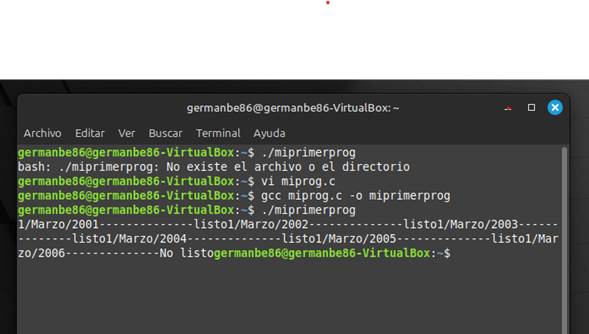
volver a ejecutar “: x” (Este comando guarda los cambios echos en el VI.)

ahora volver a ejecutar:

primero: gcc miprog.c -o miprimerprog

Luego: ./miprimerprog

¿Cual es la salida ahora?

El programa compilo correctamente y se ejecuto por pantalla en la consola.

## **9. VARIABLES DE ENTORNO**

### **9.1.** ¿Qué son las variables de entorno y para qué sirven?

Las variables de entorno son valores dinámicos que afectan los programas o procesos que se ejecutan en un servidor.

**9.1.1.**  Escriba el contenido y explique el significado de las siguientes variables: HOME / LOGNAME / PATH / HOSTNAME / IFS

· **HOME**:

* **Contenido**: El directorio de inicio del usuario actual. Cada usuario en un sistema tiene un directorio de inicio donde puede almacenar archivos y configuraciones personales.
* **Significado**: Define el directorio base de trabajo del usuario. Cuando se ejecuta el comando cd sin argumentos, el sistema cambia a este directorio.

· **LOGNAME**:

* **Contenido**: El nombre del usuario que ha iniciado sesión.
* **Significado**: Esta variable almacena el nombre del usuario actualmente autenticado en la sesión, útil para scripts y procesos que necesitan saber qué usuario está interactuando con el sistema.

· **PATH**:

* **Contenido**: Una lista de directorios separados por : que el sistema busca para encontrar ejecutables cuando se ejecutan comandos.
* **Significado**: Define la ruta que el sistema sigue para buscar los comandos o scripts que el usuario ejecuta. Si un programa no está en uno de los directorios especificados en PATH, no se podrá ejecutar sin proporcionar su ruta completa.

· **HOSTNAME**:

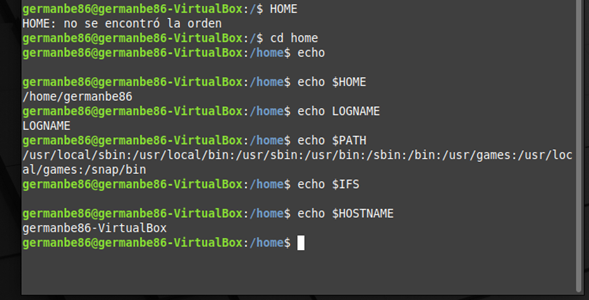
* **Contenido**: El nombre del host o máquina en la que está trabajando el usuario.
* **Significado**: Identifica de forma única la máquina en la red. Es útil cuando se trabaja con varias máquinas o servidores para distinguir entre ellos.

· **IFS** (Internal Field Separator):

* **Contenido**: El conjunto de caracteres que el shell utiliza para separar palabras al analizar la entrada de usuario. Por defecto suele ser un espacio en blanco, una tabulación o un salto de línea.
* **Significado**: Define cómo el shell separa elementos en scripts y comandos. Cambiar esta variable puede afectar cómo se procesan las listas o entradas del usuario.

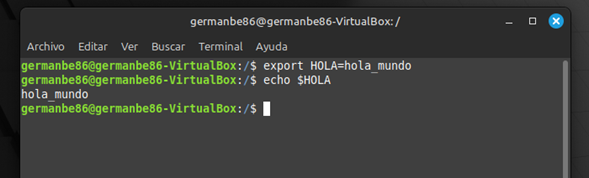
**9.1.2.**  ¿Qué comando usó para ver el contenido de las variables del punto anterior?

Para ver el contenido de las variables de entorno mencionadas en el punto anterior, se utiliza el comando **echo** seguido del nombre de la variable, precedido por un signo de dólar



**9.1.3.**  Cree una variable de entorno HOLA que contenga el mensaje “Hola mundo”.

* **export**: Es el comando que permite que la variable creada esté disponible para todos los procesos hijos del shell actual, convirtiéndola en una variable de entorno.
* **HOLA="Hola mundo"**: Asigna el valor "Hola mundo" a la variable HOLA.



La variable estará disponible solo durante la sesión actual del shell. Si cierras el terminal, se perderá. Para hacerla persistente, debes añadir el comando al archivo de configuración del Shell correspondiente.

**9.1.4.**  ¿Cuál es el uso que le da el sistema a la variable PATH? ¿Qué ocurre si intenta

ejecutar un comando que no se encuentra ubicado en alguno de los directorios que contiene la variable? ¿Cómo lo soluciona?

La variable de entorno PATH se utiliza para definir una lista de directorios en los que el sistema operativo busca programas ejecutables cuando se introduce un comando en la terminal. Cada vez que se ejecuta un comando, el sistema revisa en estos directorios (listados en la variable PATH) para encontrar el archivo correspondiente que debe ejecutarse.

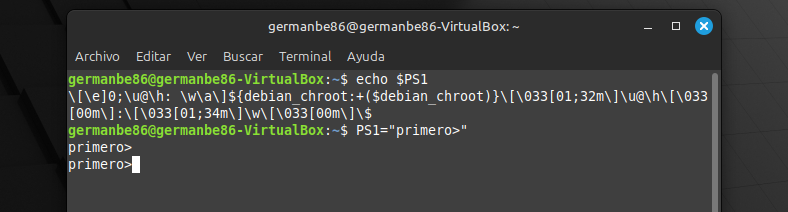
Si intentas ejecutar un comando que no se encuentra en ninguno de los directorios especificados en la variable PATH, el sistema devolverá un error indicando que no puede encontrar el comando.

**9.1.5.**  ¿Por qué existen las variables PS1 y PS2? ¿Qué es un comando multilínea?

PS traduce a “Prompt String”

#### **1.** Variable **PS1** (Prompt String 1)**:**

#### PS1 define la apariencia del prompt principal en Bash. Es el texto que se ve cuando la terminal está esperando que escribas un comando.



#### **2.** Variable **PS2** (Prompt String 2)**:**

PS2 define el prompt secundario, que aparece cuando se está ingresando un comando multilínea. Es decir, cuando un comando es largo y se necesita escribir en varias líneas.

### 

### 

### **Comando Multilínea:**

Un comando multilínea en Linux es un comando que se extiende a través de varias líneas en la terminal. Esto se puede lograr usando el carácter de continuación de línea \ al final de cada línea para indicar que el comando sigue en la siguiente línea. Es útil para comandos largos o para mejorar la legibilidad de scripts complejos.

# 

—---------------------------------------------------------------------------------------------------------------------------------------

# **Conclusiones finales:**

Este trabajo nos ha proporcionado una base sólida en el uso de Linux, desde la administración de usuarios y terminales hasta la edición de archivos y configuración del entorno. Estas habilidades son fundamentales para una gestión eficaz del sistema y nos preparan para el dia de mañana, donde debamos utilizar Linux dentro de nuestro espacio de trabajo.

Personalmente, nos parece una herramienta muy útil, tanto por su capacidad de adaptación como por su potencial para ofrecer un control profundo sobre el entorno de trabajo. Linux no solo proporciona un entorno robusto y seguro, sino que también permite una personalización extensiva que puede ser adaptada a diversas necesidades y preferencias. La experiencia de trabajar con Linux nos ha mostrado cómo se pueden realizar tareas complejas de manera eficiente, lo cual es crucial en rubros donde siempre se debe estar a la vanguardia tecnológica.

Esta base de conocimientos y habilidades adquiridas amplía nuestras competencias técnicas ya que donde el dominio de Linux puede ser un activo valioso y diferenciador. Es una habilidad que nos brinda una ventaja significativa, ofreciendo herramientas poderosas para la administración de sistemas y la resolución de problemas en un entorno dinámico y profesional.