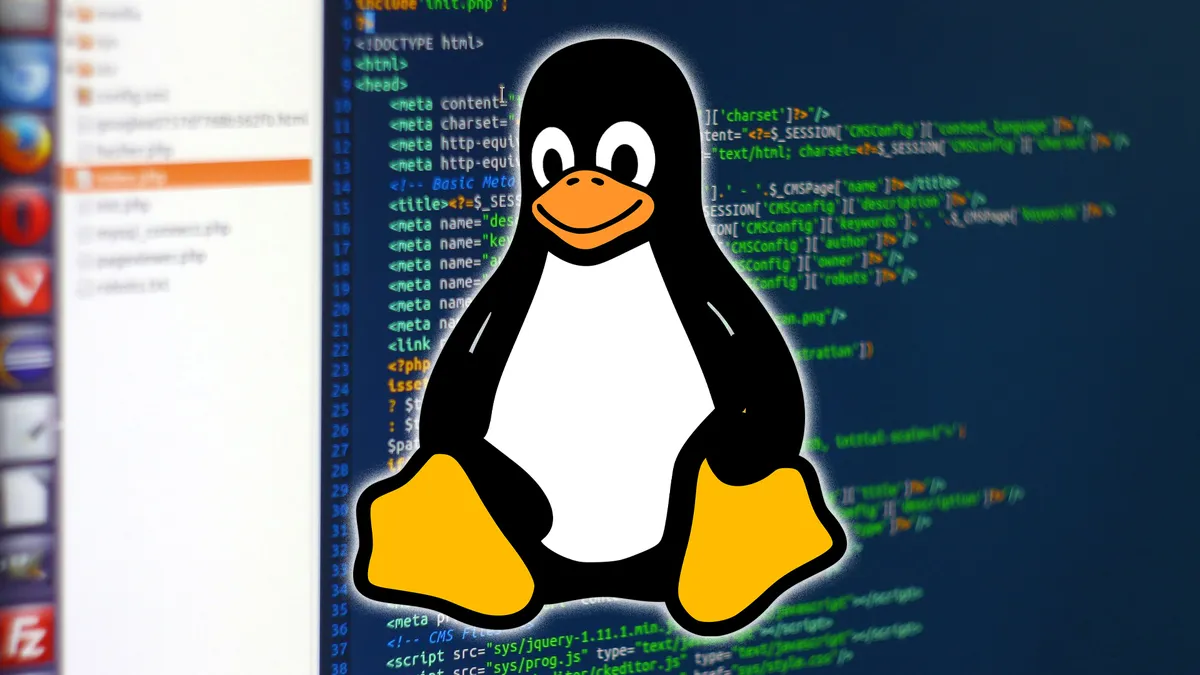
UTN | FRGP

Arquitectura de Sistemas Operativos

Trabajo Práctico GNU/Linux



Grupo 91

Integrantes:

* Diego Salas - Legajo: 30591
* Emanuel Celis - Legajo: 30284
* Facundo Díaz - Legajo: 30314
* Natalia Mucci - Legajo: 30490

**INDICE**

[OBJETIVOS 3](#_Toc165709755)

[PLAN DE TRABAJO 4](#_Toc165709756)

[DESARROLLO 5](#_Toc165709757)

[**Cuestionario** 5](#_Toc165709758)

[**Respuestas** 11](#_Toc165709759)

[1. INTRODUCCIÓN 11](#_Toc165709760)

[2. TECLADO / TERMINALES 11](#_Toc165709761)

[3. AYUDA 12](#_Toc165709762)

[4. DIRECTORIOS 14](#_Toc165709763)

[5. ARCHIVOS 15](#_Toc165709764)

[6. PERMISOS 16](#_Toc165709765)

[7. FILTROS 19](#_Toc165709766)

[8. VI 20](#_Toc165709767)

[9. VARIABLES DE ENTORNO 23](#_Toc165709768)

[CONCLUSIÓN 25](#_Toc165709769)

[BILIOGRAFÍA 26](#_Toc165709770)

[ANEXO 1 27](#_Toc165709771)

# OBJETIVOS

Para la realización del presente trabajo se plantearon los siguientes objetivos:

* Familiarizarnos con el uso de un Sistema Operativo distinto al común denominador
* Disponer de un ambiente de ejecución multitarea y multiprogramado para realizar las prácticas obligatorias y conocer los principales comandos de GNU/Linux.
* Utilizar diferentes medios para acceder a la terminal de comandos de Linux.
* Aprender a trabajar y a coordinar acciones con personas desconocidas, de manera virtual.

# PLAN DE TRABAJO

Para la realización de este trabajo, se utilizaron diferentes medios para acceder a la terminal Linux, pero todos con la distro de Ubuntu:

* Virtual Box
* Pendrive booteable
* Página Web <https://distrosea.com/>

Se utilizó el cuestionario proporcionado por la cátedra y se fueron contestando uno por uno los puntos solicitados, de acuerdo con los resultados obtenidos.

Al final de este se plantea una pequeña conclusión sobre nuestra experiencia con la terminal Linux y sus comandos.

# DESARROLLO

## **Cuestionario**

1. **INTRODUCCIÓN**
   1. ¿Qué es la cuenta de superusuario (root) y para qué se utiliza?
   2. Ingresar al sistema como superusuario (root), y realizar los siguientes pasos:
   3. adduser <apellido> (reemplazar <apellido> por el suyo).
   4. passwd <apellido> (Ingrese una contraseña (password) a su elección).
   5. logout
   6. Indique claramente qué efectuaron estos comandos, e indique qué archivo/s fueron modificados (Dentro del directorio /etc).
   7. ¿En qué directorio se encuentran los comandos utilizados en los puntos 1.3 y 1.4?
2. **TECLADO / TERMINALES**
   1. ¿Qué sucede si tecleo cat /e <tab> p <tab>? (donde tab es la tecla tabulación). Presione <tab> nuevamente ¿Qué pasó ahora?
   2. ¿Qué sucede si tecleo cat /e <tab> pas <tab>?
   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:
      1. who
      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.
      3. Ejecute nuevamente el comando who. ¿Qué diferencias encuentra con la primera vez que lo ejecutó?
      4. Ejecute el comando who am i ¿qué muestra?, ¿Qué diferencias tiene con el comando ejecutado en el punto anterior?
      5. Repita el paso 2.3.2 y el 2.3.3 hasta que no encuentre ninguna sesión para abrir.
      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).
      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).
3. **AYUDA**
   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 especifico tópico con un determinado nivel de detalle.
      1. Ingrese a info y responda:
         1. ¿Cómo se llama el nodo raíz de Info?
         2. Ubique el cursor en la línea (\* cp:) y presione ENTER.
         3. ¿Qué sucedió?
         4. ¿Cómo se llama este nodo?
         5. ¿Cuál es el próximo nodo?
         6. ¿Cómo puedo moverme al próximo nodo?
         7. ¿Cómo puedo moverme al nodo anterior?
      2. Presione la tecla 'u'.
         1. ¿Qué sucedió?
         2. ¿En qué nodo se encuentra?
      3. Repita el punto 3.1.2. hasta que llegue a la raíz de Info.
         1. ¿Con qué tecla puedo volver directamente a este nodo?.
         2. ¿Cuál es el método directo para acceder al nodo cp? (tip: sin desplazar el cursor).
      4. ¿Cómo puedo buscar una palabra clave dentro de un nodo?
      5. ¿Cómo puedo buscar la siguiente palabra clave, buscada anteriormente?
      6. ¿Cómo puedo salir de Info? -hágalo.
   2. MAN: man es un programa que formatea y muestra las páginas del manual.
      1. ¿Cuál es la diferencia entre man e info?
      2. ¿Cómo puedo ver la información de un determinado comando?
      3. ¿Cómo puedo buscar una palabra clave dentro de la página del manual?
      4. ¿Cómo puedo salir?
      5. ¿Cómo hago para buscar una palabra clave determinada en todas las páginas del manual?
      6. ¿Qué es lo sucede al realizar lo siguiente?
         1. man
         2. man man
         3. man cp
         4. man printf
         5. man fprintf
         6. man sprintf
         7. man 3 printf
      7. Si los comandos del punto anterior le dieron un mensaje de error, instale el paquete manpages-dev y vuelva a ejecutarlos (utilice sudo apt install manpages-dev si está usando Ubuntu).
      8. Del punto anterior, responder:
         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).
         2. ¿Cómo puedo invocar al man para ver directamente la función printf del lenguaje C?.
   3. HELP: help es la ayuda que ofrece el shell de GNU/LINUX para utilizar sus comandos.
      1. ¿Cuál es la diferencia entre help e info?.
      2. ¿Cuál es la diferencial entre help y man?.
      3. ¿Qué sucede al invocar al help?.
      4. ¿Cómo puedo ver la información de un determinado comando?
      5. ¿Qué sucede al utilizar el comando help cd?
   4. whereis
      1. ¿Qué sucede al utilizar el comando whereis cp?
      2. ¿Qué es la información que se muestra por pantalla al ejecutar el punto anterior?
      3. ¿Qué ocurre si se ejecuta whereis \* sobre un directorio? (**Tip**: si no pasa nada, intentelo nuevamente pero primero ejecute cd /bin)
      4. ¿Cuál es la diferencia entre whereis y find?
   5. whatis
      1. ¿Qué sucede al utilizar el comando whatis cp?
      2. Cambie al directorio /bin, y ejecute el comando whatis \* ¿Qué ocurrió?
      3. Utilice el comando apropos passwd y whatis passwd. Enumere las diferencias encontradas en el resultado de cada uno de los comandos.
4. **DIRECTORIOS**
   1. ¿Para qué se usa el comando cd? Ejecute las siguientes variantes de cd e indique cuál fue el resultado obtenido:
      1. cd /
      2. cd
      3. cd /etc
      4. cd..
      5. cd ..
   2. Bash sobre directorios:
      1. ¿Cuál/es son las diferencias entre el path absoluto y el path relativo?
      2. ¿Qué es lo que realizan las siguientes operaciones? (**tip:** si no encuentra la diferencia primero haga cd /, y luego vuelva a intentar)
         1. cd ~
         2. cd -
      3. ¿Cuál es la diferencia entre cd ~ y cd -?
   3. Operaciones con directorios:
      1. ¿Con qué comando se puede crear un directorio?
      2. ¿Con qué comando se puede borrar un directorio?
      3. ¿Qué sucede sí el directorio no está vacío?
      4. ¿Cómo puedo salvar la situación anterior? (Sin borrar uno a uno los archivos existentes).
   4. ¿Qué significa la expresión ./ cuando se utiliza delante de un archivo? ¿Para que sirve?
   5. ¿Cómo puede moverse entre directorios sin utilizar el PATH completo?
5. **ARCHIVOS**
   1. ¿Qué hacen los siguientes comandos?
      1. cp
      2. mv
      3. rm
      4. rcp
      5. rsh
      6. scp
      7. ssh
   2. ¿Con qué comando puedo concatenar el contenido de dos archivos?
      1. ¿Se puede usar ese comando para otra cosa?
   3. Haga un ls –l /dev
      1. ¿Qué significa el primer carácter?
      2. ¿Cuáles son todos los posibles valores que puede contener ese campo y que significa cada uno?
   4. ¿Para qué sirve el comando touch? ¿qué utilidad le encuentra?
6. **PERMISOS** 
   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)
   2. ¿qué significan cada caracter? ¿cómo están agrupados?
   3. ¿Cómo se asignan los permisos? (detalle los comandos).
   4. ¿Qué son el owner, y el group de un archivo? ¿Se pueden cambiar?
   5. Intente cambiar los permisos de un archivo perteneciente al root (sesionado como usuario). Explique qué sucedió.
   6. Explique la forma de cambiar los permisos con valores en octal.
   7. ¿Cuál es el significado de los permisos en los directorios (se debe indicar que indica una r, una w, y una x)?
7. **FILTROS**
   1. ¿Cuál es la diferencia de los comandos more, less y cat? De un ejemplo de cada uno.
   2. ¿Cuál es la diferencia entre tail y head?
   3. ¿Qué es lo que realiza el comando uniq?
   4. ¿Qué es lo que realiza el comando grep?
8. **VI**
   1. Ejecute la siguiente instrucción: vi $HOME/prueba.txt ¿Qué sucedió? Ahora ejecute todos los pasos detallados a continuación.
      1. Escriba la siguiente frase: “Este es el archivo prueba.txt de <nombre y apellido>”
      2. ¿Qué tuvo que hacer para poder escribir la frase?
      3. Guarde el archivo, y salga del editor. ¿Qué comando utilizó?
      4. Ingrese nuevamente al archivo.
      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>”

* + 1. Describa todos los pasos que tuvo que realizar.
    2. Guarde el archivo y continúe la edición. ¿Qué comandos utilizó?
    3. Borre la línea de “Legajo”. Indique por lo menos dos formas de realizarlo.
    4. Invierta el orden de las líneas “Curso y TP”. No está permitido rescribirlas. ¿Qué comandos utilizó?
    5. Ubíquese en la línea 2 (dos) del archivo. No está permitido usar las teclas del cursor, ni el mouse. ¿Qué comando utilizó?
    6. Marque para copiar las líneas 2, 3, y 4 (todas juntas, no de a una a la vez). ¿Cómo lo realizó?
    7. Ubíquese al final del archivo (sin usar las teclas del cursor), y pegue dos veces el contenido del buffer. ¿Qué comando usó?
    8. Deshaga uno de los copiados. No está permitido borrar línea por línea, ni carácter a carácter. ¿Qué comando usó?
    9. ¿Cómo busco la palabra “Documento”? ¿Cómo busco la segunda ocurrencia de una palabra?
    10. ¿Cómo puedo reemplazar la palabra “Documento” por “Documento:” (sin borrar, o realizar el reemplazo a mano)’
    11. Guarde el archivo y salga.
    12. 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).

./miprimerprog

¿Cuál es la salida?

volver a ejecutar vi miprog.c

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

Corrija los errores que surgieron al utilizar el comando gcc.

volver a ejecutar: x

ahora volver a ejecutar:

gcc miprog.c -o miprimerprog

./miprimerprog

¿Cual es la salida ahora?

1. **VARIABLES DE ENTORNO**
   1. ¿Qué son las variables de entorno y para qué sirven?
      1. Escriba el contenido y explique el significado de las siguientes variables: HOME / LOGNAME / PATH / HOSTNAME / IFS
      2. ¿Qué comando usó para ver el contenido de las variables del punto anterior?
      3. Cree una variable de entorno HOLA que contenga el mensaje “Hola mundo”.
      4. ¿Cuál es el uso que le da el sistema a la variable PATH? ¿Qué ocurres si intenta ejecutar un comando que no se encuentra ubicado en alguno de los directorios que contiene la variable? ¿Cómo lo soluciona?
      5. ¿Por qué existen las variables PS1 y PS2? ¿Qué es un comando multilínea?

## **Respuestas**

### 1. INTRODUCCIÓN

**1.1.** La cuenta de superusuario **root** **es el administrador del sistema**. Él tiene acceso completo a todos los archivos y recursos del sistema y además cuenta con los permisos para cualquier tarea. Se utiliza para realizar tareas de como instalar software, modificar configuraciones del sistema, y administrar usuarios y permiso Los usuarios comunes en cambio no tienen permisos para instalar programas o escribir en los directorios del sistema.

**1.2**. Para ingresar al sistema como superusuario (root), generalmente se utiliza el comando **su** seguido de la contraseña de root. Pero cuando utilizamos el pendrive bootebale o la página web, como se eligió la opcion de probar Linux y no instalarlo, no nos permitó logearnos como superusuario, pero si simular las acciones mediante el comando **sudo** en vez de **su.**

**1.6.** El comando **adduser** agregó un usuario al sistema. El comando **passwd** asignó una contraseña al usuario creado anteriormente. Después de confirmar la contraseña, el sistema nos pidió completar datos adicionales, correspondientes al usuario que estábamos creando: Full Name, Room Number, Work Phone, Home Phone, Other. Una vez que los datos están completos, pregunta si los datos son correctos y después de responder Y ya crea el nuevo usuario.

Los archivos modificados fueron:

* **etc/passwd,** que contiene información sobre las cuentas de usuario
* **etc/shadow,** que contiene las contraseñas encriptadas de los usuarios.
* **etc/subuid**
* **etc/subguid**
* **etc/group**

**1.7**. Los comandos utilizados en el punto 1.3 y 1.4 se encuentran en el directorio **usr/sbin.**

### 2. TECLADO / TERMINALES

**2.1**. Al hacer el primer <tab> (después de teclear cat /e) se auto-completa el comando con el directorio **/etc**. Al hacer el segundo <tab> lista las carpetas y archivos que comiencen con la letra **p**.

**2.2.** Se autocompleta con la siguiente ruta: **/etc/passwd**

**2.3.1**. Muestra el usuario logueado, y también muestra la fecha y la hora que hizo login.

**2.3.3**. La diferencia es que esta vez muestra otro login con el mismo usuario, pero en **tty3**.

**2.3.4.** Muestra el usuario logueado en **tty3**, y también muestra la fecha y la hora que lo hizo. La diferencia con el comando anterior es que ahora solo muestra el usuario logueado en **tty3**.

**2.3.6.** Los distintos tipos de terminales se denominan terminales virtuales. En cada uno de estos terminales puede mantenerse una sesión de trabajo distinta. La palabra **tty** usada para designar a los terminales viene de **T**ele**TY**pe (teletipo). Para cambiar de un terminal virtual a otro tenemos que pulsar **<Ctrl><Alt><F3>**, **<Ctrl><Alt><F4>**, **<Ctrl><Alt><F5>** ó **<Ctrl><Alt><F6>** con esto tendremos acceso a los terminales **/dev/tty3**, **/dev/tty4**, **/dev/tty5** ó **/dev/tty6** respectivamente.

### 

### 3. AYUDA

**3.1. INFO**

**3.1.1.1.** Se llama nodo raíz de Info se llama Top.

**3.1.1.3.** Al ubicar el cursor en la línea (\* cp:) y presionar ENTER, se abre la página de información correspondiente al comando **cp**.

**3.1.1.4**. Este nodo se llama **cp Invocation**.

**3.1.1.5.** El próximo nodo es **dd Invocation.**

**3.1.1.6.** Se puede mover al próximo nodo con la letra **n** (Next).

**3.1.1.7.** Se puede mover al nodo anterior con la letra **p** (Previous).

**3.1.2.1.** Subió un nodo en el árbol de la documentación.

**3.1.2.2.** Al presionar la tecla 'u', se vuelve al nodo padre del nodo actual: **Basic operations.**

**3.1.3.1**. Con la tecla **d** se puede volver directamente al nodo raíz.

**3.1.3.2.** Si se está dentro de la pantalla Info, se puede escribir el comando **go** y después el nombre del nodo al que se quiere ir, en este caso sería **cp Invocation.**

Si estamos en la pantalla de la terminal, directamente se escribe el comando **info cp**

**3.1.4.** Se puede buscar una palabra clave escribiendo la letra **s**, luego la palabra a buscar y presionamos **ENTER**.

**3.1.5.** Para buscar la siguiente palabra clave, buscada anteriormente, se presiona la letra **s** seguido de la tecla **ENTER**.

**3.1.6.** Se puede salir de **info** con la letra **q**.

**3.2. MAN**

**3.2.1.** La principal diferencia entre man e info, radica en la forma y en la cantidad de información que proporcionan.

* **man** (abreviación de manual) muestra información más concisa y rápida en formato de páginas de manuales lineales. La información está organizada en secciones numéricas.
* **info** tiene información más detallada y estructurada, en forma de nodos entrelazados e interactivos.

**3.2.2.** Para ver la información de un comando específico, se escribe **man** seguido del nombre del comando.

**3.2.3** Si se está dentro del manual, para buscar una palabra clave, se escribe el signo de pregunta **?** y después la palabra que se quiere buscar. Sino directamente en la consola se escribe el comando **apropos** seguido de la palabra clave.

**3.2.4** Al igual que con info, pra salir de man hay que apretar la tecla **q** de quit.

**3.2.5** Para buscar una palabra clave determinada en todas las páginas del manual, en la pantalla principal de la terminal hay que escribir el comando **man -K** seguido de la palabra clave.

**3.2.6.1.** **man:** muestra la página de manual del comando man.

**3.2.6.2**. **man man:** muestra la página de manual del comando man.

**3.2.6.3**. **man cp:** muestra la página de manual del comando cp.

**3.2.6.4.** **man printf:** muestra la página de manual del comando printf.

**3.2.6.5.** **man fprintf:** muestra la página de manual del comando fprintf.

**3.2.6.6. man sprintf:** muestra la página de manual del comando sprintf, que es la misma que fprintf.

**3.2.6.7.** **man 3 printf:** muestra la página de manual de la función printf en la sección 3 del manual, que son las funciones de la biblioteca (que es la misma página que las dos anteriores)

**3.2.8.1** Al invocar **man** junto con **fprintf y sprintf** muestra la misma página ya que fprints y sprintf, son funciones de la biblioteca C, por eso están ambos en la misma sección del manual, sección 3 - funciones de la biblioteca. En cambio, al invocar **man** con el comando **printf** se muestra una página diferente, porque printf es un comando del sistema y se encuentra en la sección 1 -comandos de usuario.

**3.2.8.2** Para invocar man para ver directamente la función printf del lenguaje C se utiliza el comando **man 3 printf.**

**3.3. HELP**

**3.3.1. help** proporciona ayuda específica para los comandos del shell de GNU/LINUX, mientras que **info** ofrece documentación más detallada y estructurada sobre una gran variedad de temas, no solo sobre los comandos, sino también relacionados al sistema operativo.

**3.3.2.** En cuanto a la diferencia entre **help** y **man**, se basa en el tipo de información que da cada comando. help proporciona información sobre los comandos del shell y su uso, mientras que man muestra páginas de manual sobre comandos, funciones y archivos del sistema.

**3.3.3.** Al invocar **help,** se muestra una lista de los comandos disponibles y una pequeña descripción de cada uno. Si un comando no aparece en help, se lo puede buscar en **man** o en **info**

**3.3.4.** Para ver la información de un comando específico hay que escribir **help** seguido del nombre del comando.

**3.3.5.** El comando **help cd**, muestra una descripción del comando cd junto con su sintaxis y opciones de uso.

**3.4.1.** El comando **whereis cp,** muestra la ubicación de los archivos binarios, código fuente y páginas de manual relacionadas con el comando cp.

**3.4.2.** La información que se muestra por pantalla al ejecutar el punto anterior incluye la ubicación del archivo ejecutable, el código fuente y las páginas de manual relacionadas con el comando cp.

**3.4.3.** Al ejecutar **whereis** \* sobre un directorio, se muestra la ubicación de los archivos binarios, código fuente y páginas de manual de **todos** los comandos en ese directorio.

**3.4.4.** La diferencia principal entre **whereis y find** es que **whereis** busca la ubicación de los archivos binarios, código fuente y páginas de manual relacionados con un comando específico, mientras que **find** busca archivos en el sistema de archivos en general.

**3.5.1.** El comando **whatis cp**, muestra una descripción muy abreviada del comando cp.

**3.5.2.** Al cambiar al directorio /bin y ejecutar el comando whatis \*, se muestra descripción acotada de todos los comandos en el directorio /bin.

**3.5.3.** El comando **apropos passwd** arrojó 15 resultados, ya que buscó en todas páginas del manual en donde aparece la palabra clave passwd en su descripción y devolvió la lista de comandos y funciones relacionados con la palabra.

En cambio **whatis passwd** solo muestra 3 resultados,ya que solo trae información del comando passwd en sí mismo, sin tener en cuenta comandos o funciones asociadas.

### 4. DIRECTORIOS

**4.1.** El comando cd se utiliza para cambiar el directorio de trabajo actual.

**4.1.1.** Ingrese al directorio raíz.

**4.1.2.** Volvió al directorio principal.

**4.1.3.** Ingrese al directorio de configuración de sistemas.

**4.1.4.** Salió el mensaje “command not found”.

**4.1.5**. Regrese al directorio raíz.

**4.2. Bash sobre directorios:**

**4.2.1** La diferencia es el punto de referencia desde el cual se indica la especificación de la ubicación del directorio. El path absoluto representa la ruta completa del recurso, parte del directorio raíz hasta llegar al archivo buscado y el path relativo representa solo una parte de la ruta, ya que en el se tiene en cuenta el directorio actual desde el que se está trabajando.

**4.2.2 / 4.2.2.1 / 4.2.2.2**. Ingrese cd / primero, luego ingrese cd ~ regrese al directorio principal o directorio inicio y cuando ingrese cd – regrese al directorio raíz.

**4.2.3.** La diferencia es que el comando cd ~ cambia al directorio principal o directorio de inicio del usuario y el comando cd – cambia al directorio anterior del directorio actual.

**4.3. Operaciones con directorios:**

**4.3.1**. Se puede crear un directorio con el comando “ mkdir “.

**4.3.2**. Se puede borrar un directorio con el comando “ rm -d “.

**4.3.3.** Si el directorio no está vacío saldrá un mensaje de error.

**4.3.4.** Se utiliza el comando “ rm –r “.

**4.4.** Indica que el archivo se encuentra en el directorio actual en el que nos encontramos, sirve para referenciar archivos o directorios en el mismo directorio sin tener que especificar la ruta completa.

**4.5.** Para moverse entre directorios sin utilizar el PATH completo se utiliza el comando “cd” y el nombre del directorio a moverse.

### 5. ARCHIVOS

**5.1.1.** El comando **cp** permite copiar archivos y directorios, y ubicarlos en el destino deseado.

**5.1.2**. El comando **mv** se utiliza para mover o cambiar el nombre del archivo o directorio. Se pueden mover varios a la vez.

**5.1.3.** El comando **rm** se utiliza para eliminar archivos y directorios de forma permanente.

**5.1.4.** El comando **rcp** es una herramienta de copia remota, copia archivos o directorios entre un sistema local y un sistema remoto.

**5.1.5.** El comando **rsh** permite ejecutar comandos en ordenadores remotos de forma directa.

**5.1.6.** El comando **scp** permite realizar la copia de archivos o directorios de forma segura con un sistema remoto.

**5.1.7.** El comando **ssh** se utiliza para conectarse a un servidor remoto de forma segura a través de internet. Es un protocolo que permite a los usuarios acceder y administrar servidores remotos como si estuvieran trabajando en ellos localmente. El comando ssh requiere una clave segura para autentificar la conexión, que se genera automáticamente cuando se conecta por primera vez al servidor.

**5.2.** El comando **cat** sirve para concatenar el contenido de dos archivos.

**5.2.1.** Si, este comando sirve también para mostrar, concatenar y crear archivos.

**5.3.1.** El primer caracter indica el tipo de archivo.

**5.3.2.** Los posibles valores son:

* **‘-’ :** (Regular file) Un archivo regular
* **‘d’ :** (Directory) Un directorio
* **‘b’ :** (Block device) Un archivo especial en bloque
* **'c’ :** (Chararter device) Un archivo especial en caracteres
* **'l’ :** (Symbolic link) Apunta al archivo o directorio original a travez de la ruta de acceso
* **'p’ :** (Named pipe) Archivo especial que permite que los datos se envíen entre procesos de manera unidireccional
* **'s’ :** (Socket) Permite que dos procesos se comuniquen entre sí

**5.4.** El comando **touch** se usa para crear archivos vacíos o si existe el archivo modifica la fecha y el horario de creación.

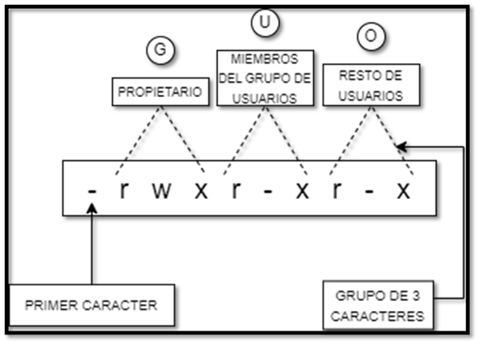
La utilidad que le encuentro a esta función es para proyectos que han estado inactivos por un tiempo, que al actualizarla, a la hora de retomar el proyecto, poder encontrarlos con mayor rapidez.

### 6. PERMISOS

**6.1.** Los siguientes 9 caracteres indican los permisos del archivo otorgados a los 3 grupos.

**6.2.** Los caracteres están agrupados en tres bloques de tres caracteres cada uno:

* El primer bloque, Owner, son los tres primeros caracteres y corresponde a los permisos de dueño del archivo.
* Segundo bloque, Group, son los siguientes tres caracteres e indican los permisos del grupo
* Tercer bloque, Others, son los últimos tres caracteres e indican los permisos para otros usuarios.



Estos 3 caracteres y tiene un orden predeterminado:

* Primero **‘r’**: lectura
* Segundo **‘w’**: Escritura
* Tercero **‘x’** Ejecución

Los guiones medios (-) que se encuentran reemplazando caracteres significan que el permiso de esa posición no está otorgado al grupo determinado.

Por ejemplo, en la imagen de ejemplo el propietario tiene todos los permisos otorgados, pero su grupo y el resto de usuarios solo tienen permiso de lectura y ejecución.

**6.3.** Los permisos se asignan con el comando **chmod**, seguido de una combinación de letras y/o símbolos, que representan las entidades, los permisos, y el nombre del archivo o directorio al que se le están asignando los permisos.

La combinación de letras y/símbolos depende de:

* Tipo entidad:
* Propietario (owner): u
* Gupo (group): g
* Otro(other): o
* Si se quiere agregar permiso: +
* Si se quiere quitar permiso: -
* Si se quieren establecer permisos exactos: =
* Que permisos se quieren agregar/quitar/establecer:
* Lectura(read): r
* Escritura (write): w
* Ejecución (execution): x

**Ejemplos:**

* Si le quiero dar solo permisos de lectura y escritura al grupo del archivo:

**chmod g+rw nombre del archivo**

* Si le quiero quitar permiso de ejecución a otros usuarios

**chmod o-x nombre del archivo**

* Si quiero establecer todos los permisos para el propietario y nada para el grupo y otros usuarios:

**chmod u=rwx, go= nombre del archivo**

**6.4. El** **owner** (propietario) de un archivo es el usuario que lo creó o el usuario que lo posee. El **group** (grupo) de un archivo es el grupo al que pertenece el propietario del archivo.

Los dos se pueden cambiar utilizando los comandos **chown** para el usuario y **chgrp** para el grupo.

**6.5.** Al intentar cambiar los permisos de un archivo perteneciente al root (sesionado como usuario), sale una leyenda de **"Operation not permitted”,** porque los usuarios comunes no podemos modificar archivos que le pertenecen al superusuario.

**6.6.** Para cambiar los permisos utilizando valores en octal, se utiliza el comando **chmod** seguido de un número octal que representa el permiso. Cada permiso se representa con un número de tres dígitos y cada dígito representa un grupo de permisos: el propietario, el grupo y otros.

Para calcular cada uno de los dígitos, se suman los valores correspondientes a los permisos que se desean otorgar y los valores son:

* r (Lectura): 4.
* w (Escritura): 2.
* x (Ejecución): 1.

Ejemplo:

**chmod 777 archivo**

Tanto el usuario (primer 7), como el grupo (segundo 7) como los otros usuarios (último 7), tienen permisos de lectura, escritura y ejecución (4+2+1)

**6.7** En los directorios, los permisos tienen el siguiente significado:

* La **r** indica que el usuario tiene permiso de lectura en el directorio, puede ver el contenido del directorio.
* La **w** indica que el usuario tiene permiso de escritura en el directorio, puede crear, eliminar o modificar archivos en el directorio.
* La **x** indica que el usuario tiene permiso de ejecución en el directorio, puede acceder al contenido del directorio y ejecutar comandos en él.

### 7. FILTROS

**7.1**. Los tres comandos permiten ver archivos por la pantalla, pero cada uno tiene sus particularidades:

* **More:** Muestra el contenido de un archivo de texto por páginas, también permite desplazarse por el mismo, página por página, utilizando las teclas de dirección del teclado hacia arriba y hacia abajo o la barra espaciadora, también permite desplazarse línea por línea utilizando la tecla enter. En la parte inferior se muestra el porcentaje de avance en el archivo.

Se puede utilizar la letra ‘q’ para salir o cuando se llega a la última página (100%) automáticamente se vuelve a la línea de comando.

* **Less: e**ste comando muestra el contenido de un archivo de texto por páginas, permite desplazarse por el mismo con las teclas de dirección en las cuatro direcciones, también se puede utilizar la tecla de enter para desplazarse line por línea y la funcionalidad que lo destaca es la de buscar una cadena de caracteres deseada utilizando la tecla `/` y luego la cadena de caracteres que deseas buscar en el documento. Luego de utilizar la función de ‘/’ puedes utilizar las letras ‘n’ para desplazarte hacia la siguiente vez que aparece la palabra buscada a través del archivo y ‘N’ para desplazarte hacia la anterior.

Para salir hay que apretar la letra q, ya que no tiene salida automática al final.

* **cat** muestra el contenido completo de un archivo de texto en una sola pantalla, pero no puedes desplazare a través de este.

**7.1** Ejemplos:

* Escribir “cat prueba.txt”, esto mostrara el archivo de texto llamado “prueba” de forma completa. Ver imagen1 en Anexo1
* Escribir “more prueba.txt” y presionar la tecla de dirección hacia abajo hasta desplazarnos hasta la el final del archivo. Ver imagen 2 en Anexo 1
* Escribir “less prueba.txt” y buscar página por página con la tecla barra espaciadora, desplazarnos línea por línea con enter/flecha de dirección hacia abajo o hacia arriba hasta llegar al final del archivo. Ver imagen 3 en Anexo1

**7.2** La diferencia entre head y tail es que head muestra las primeras líneas de un archivo y tail muestra las últimas.

**7.3.** El comando uniq sirve para filtrar y eliminar líneas de textos duplicadas en un archivo. Pero para que se las considere duplicadas, las líneas tienen que estar juntas.

**7.4**. El comando grep se usa para buscar texto en archivos. Se puede buscar una palabra o expresión regular en uno o varios archivos.

### 8. VI

**8.1** Con la instrucción **vi $HOME/prueba.txt** se abrió el editor de texto vi y se creó un nuevo archivo llamado "prueba.txt" en el directorio de inicio.

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

**8.1.2**. Para poder escribir la frase, primero tuve que apretar la tecla “I”, para poder entrar al modo de inserción.

**8.1.3**. Para poder guardar el archivo, primero presioné la tecla **ESC** para salir del modo inserción y después escribí la instrucción **“:wq”.**

**8.1.4**. Ingrese nuevamente al archivo.

**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**. Para resolver los puntos 8.1.4 y 8.1.5, usamos dos maneras:

**FORMA 1**

* Entrar de nuevo al archivo con la instrucción vi $HOME/prueba.txt
* Apretar la tecla “I” para entrar el modo de inserción
* Lo que se había escrito en el punto 8.2.2 se movió para abajo apretando la tecla ENTER.
* Se posiciono el cursor en la primera línea y después se escribió el texto.
* Apreté la tecla ESC para salir del modo inserción.

**FORMA 2**

* Entrar de nuevo al archivo con la instrucción vi $HOME/prueba.txt
* Sin entrar en el modo de inserción y estando posicionados en la única línea que está escrita, apretar la letra **O mayúscula**. Este comando te inserta una línea por encima del cursor y al mismo tiempo te abre el modo de inserción, por lo que ya se puede escribir. Aclaración, es importante distinguir entre **O mayúscula y o minúscula,** ya que la minúscula interna una línea debajo y lo que queremos es escribir arriba.
* Apreté la tecla ESC para salir del modo inserción.

**8.1.7**. Para guardar el archivo y continuar con la edición, una vez que salí del modo de inserción, escribí la instrucción **:w**

**8.1.8** Para borrar la línea de legajo podemos:

1. Posicionarnos con el cursor en la línea donde está el legajo y apretar **“dd”** y se elimina la línea completa (siempre fuera del modo de inserción)
2. Posicionarnos con el cursor en la línea donde está el legajo y apretar la **letra “x”** varias veces, y así se borran de a uno todos los caracteres que están en la línea (siempre fuera del modo de inserción).
3. Escribir el comando **:g/legajo/d** y apretar **Enter**.

**8.1.9** Para invertir las líneas “Curso” y “TP” hay que posicionarse en la línea donde está escrito “Curso” y apretar las teclas **dd**, que elimina la línea, pero la guarda en el buffer. Después (como es cursor queda ya posicionado en la línea de tp) apretar la tecla **p**, que pega la línea que eliminamos, debajo de donde está el cursor.

**8.1.10**. La línea 2 la pudimos ubicar de dos maneras. Saliendo el modo inserción se escribe el comando **:2** o se pueden apretar las teclas **2G.**

**8.1.11**. Para copiar las líneas 2,3 y 4, escribimos el comando **:2,4y**

**8.1.12** Para ir al final del archivo aprieto la tecla **G** y para copiar dos veces lo que tengo en el buffer aprieto dos veces la letra **p**, una por cada vez que quiero pegar.

**8.1.13** Para deshacer uno de los copiados, escribo **:u,** o aprieto la tecla **u,** que deshace el último comando.

**8.1.14**. Para buscar la palabra “documento”, fuera del modo de inserción escribimos **/documento.** Para buscar la siguiente ocurrencia de la palabra, apretamos la letra **n** y para volver a la ocurrencia anterior **N.**

**8.1.15**. Para hacer el reemplazo de la palabra, escribimos **:%s/Documento/Documento:/g**

**8.1.17**. La salida por pantalla da una serie de errores, ya que se está creando un programa en lenguaje C y no se puede ejecutar. Ver imagen 4 en Anexo 1.

Al ejecutar “:%s/\/3/\/Marzo/g” se hicieron 6 reemplazos. Se reemplazaron solo los números 3, que estaban entre / / por la palabra Marzo. No se remplazó el número 3 del año 2003.

Ver imagen 5 del Anexo 1.

Los errores que se corrigieron fueron:

* Se agregó una línea al principio del archivo **#include <stdio.h>**
* Se reemplazó **principal ()** por **int main ()**
* Se agregó **\n** al final de cada línea para se le imprima una línea debajo de otra
* Se agregó **return 0;** antes del cierre de la llave.

Ver imagen 6 del Anexo 1.

Volvemos a ejecutar gcc miprog.c -o miprimerprog ./miprimerprog y la salida fue:

1/Marzo/2001 ------------- listo

1/Marzo/2002 ------------- listo

1/Marzo/2003 ------------- listo

1/Marzo/2004 ------------- listo

1/Marzo/2005 ------------- listo

1/Marzo/2006 ------------- No listo

Ver imagen 7 del Anexo 1.

### 9. VARIABLES DE ENTORNO

**9.1.1**

•**HOME:** /home/natalia

Esta variable almacena la ruta del directorio de inicio de un usuario. La misma se utiliza cuando se inicia sesión, dependiendo del nombre del usuario que inicia, home apuntara al inicio de ese usuario e iniciara esa sesión en el directorio de inicio de ese usuario.

**•LOGNAME:** natalia

Contiene el nombre de inicio de sesión del usuario actual.

**•PATH:/**usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin:/snap/bin

La variable de entorno PATH es una variable de Linux que contiene una lista de todas las rutas en donde se encuentran los archivos ejecutables que podemos invocar.

•**HOSTNAME:** Ubuntu

Es el nombre del dispositivo actual en el que se está trabajando.

**•IFS:**

La variable de entorno IFS que significa Internal Field Separator (separador de campos internos), sirve para indicar que valor se usa como separador. Por eso es un espacio vacío.

Ver Imagen 8 del Anexo 1.

**9.1.2** Utilizamos los comandos ‘**echo’ $“variable” y ‘printenv’ “variable”.**

**9.1.3** Ingresando **export HOLA=“Hola mundo”** creamos la variable nueva y con el comando **echo $HOLA**, comprobamos su contenido.

Ver Imagen 8 del Anexo 1.

**9.1.4** PATH almacena las rutas en donde se encuentran los comandos que nosotros podemos utilizas en la consola, al nosotros escribir algún comando, por ejemplo ‘ls’.

Cuando nosotros ejecutamos el comando ‘ls’ el sistema va a buscar este comando dentro de todas las rutas que están almacenadas en la variable de entorno PATH hasta encontrar el comando que ingresamos, en este caso ‘ls’.

Si utilizamos el comando “echo $PATH | tr : \\n” podremos visualizar de forma vertical todas las rutas de la variable PATH, al ejecutar cualquier comando en la consola el sistema buscara ruta por ruta hasta encontrar el comando que deseamos ejecutar.

Si intentamos ejecutar un comando que no se encuentra en los directorios del PATH, la consola nos devolverá un mensaje de error diciendo que el comando no pudo ser encontrado.

Para solucionar esto existen dos formas:

* Podemos especificar la ruta completa donde se almacena el comando, por ejemplo, la ruta para ‘ls’ podríamos escribir ‘/usr/bin/ls’ con esto estamos especificando la ruta donde se encuentra el comando ‘ls’.
* Podemos agregar la ruta donde se encuentra nuestro comando a la variable PATH, esto puede hacerse modificando PATH, para ello utilizaremos el comando export PATH=“ruta/que/deseamos/agregar:$PATH” Es importante agregar $PATH al final para no sobrescribir todos los directorios que ya estaban guardados, si no lo hacemos podríamos afectar gravemente el funcionamiento de nuestra terminal, de no agregarlo deberíamos incluir todas las rutas prexistentes además de las que queremos agregar al PATH. Hay que tener en cuenta que esta modificación es temporal y solo se agrega a la sección actual, es decir que se pierde cuando se cierra la terminal.

**9.1.5** Las variables PS1 y PS2 existen para configurar algunos aspectos de la consola de comandos.

PS1 sirve para configurar todo lo que viene antes de la línea de comandos que deseamos ejecutar, es ese texto que vemos a la izquierda cuando estamos escribiendo los comandos, en caso de mi consola es natalia@Ubuntu:~$

PS2 sirve para configurar lo que nos retorna la consola de comandos cuando estamos ejecutando un comando que requiera que ingresemos varias líneas para terminar de ejecutarlo.

Por ejemplo, si ejecutamos el comando ‘for i in 1’ y presionamos enter la consola esperará que ingresemos más comando para completar el ciclo for, el símbolo utilizado por defecto será ‘>’.

Un comando multilínea es aquel que debe ser escrito en varias líneas, esto se debe a que algunos comandos son muy complejos de escribir en una sola línea de comandos, o que su sintaxis y legibilidad requieren que se escriba en varias líneas, sin embargo, hay algunos comandos que pueden escribirse en una sola línea en lugar de en múltiples líneas, pero suele ser poco práctico hacerlo de esta forma debido nuevamente a la complejidad, legibilidad y estructura del comando que queremos usar.

# CONCLUSIÓN

Este trabajo nos permitió conocer el sistema operativo Linux y la terminal de comandos, así como desarrollar habilidades prácticas para trabajar en este entorno.

Hemos aprendido a utilizar una variedad de comandos y herramientas para administrar el sistema, manipular archivos y directorios, y desarrollar programas en lenguaje C.

Para muchos de nosotros, acostumbrados al entorno más familiar de Windows, al principio no nos resultó fácil. Sin embargo, a medida que fuimos explorando y probando los diferentes comandos, pudimos apreciar su potencial y flexibilidad. A su vez, el trabajo en equipo y la ayuda entre compañeros, fueron clave para superar los obstáculos y avanzar en nuestro aprendizaje.

Esta experiencia nos dio una base sólida para continuar explorando y aprendiendo sobre Linux y sus aplicaciones en el futuro.

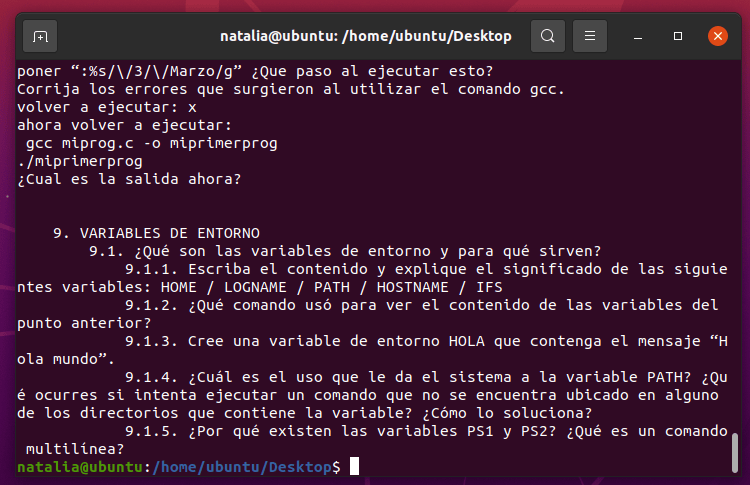
# BILIOGRAFÍA

1. Bermúdez León, M. J. (2020). Introducción a Linux <https://repositorio.usam.ac.cr/xmlui/bitstream/handle/11506/2087/LEC%20ING%20SIST%200052%202020.pdf?sequence=1&isAllowed=y>
2. Sánchez Anguix, V. (2021). Comandos básicos de Linux para trabajar con el sistema de ficheros. <https://media.upv.es/player/?id=785398f0-c232-11eb-be1f-afc4043fc67c>
3. Ciberaula. (s.f.). Curso de comandos en Linux I. <https://linux.ciberaula.com/articulo/Curso_comandos_en_Linux_I/>
4. Canonical Ltd. (s.f.). RootSudo. En Guía del escritorio de Kubuntu. <https://help.ubuntu.com/kubuntu/desktopguide/es/root-and-sudo.html>
5. FPGenRed. (s.f.). GNU/Linux: Información. https://www.fpgenred.es/GNU-Linux/info.html

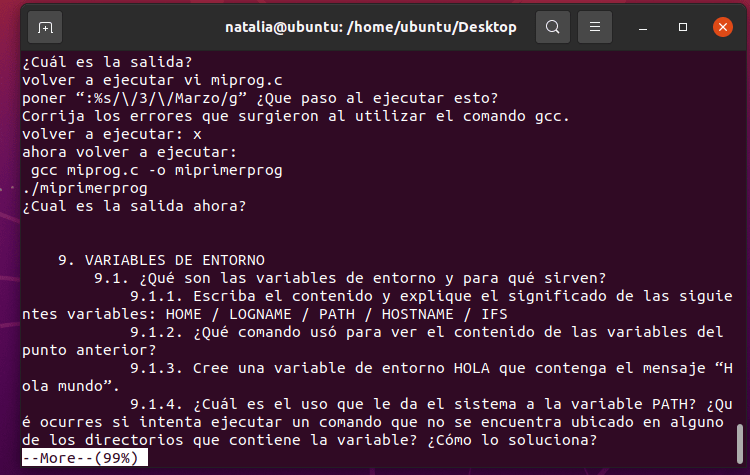
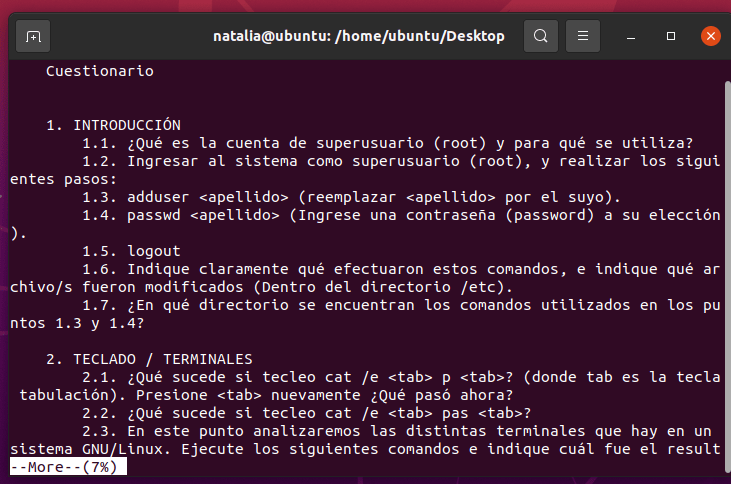
# ANEXO 1

**PUNTO 7.1**

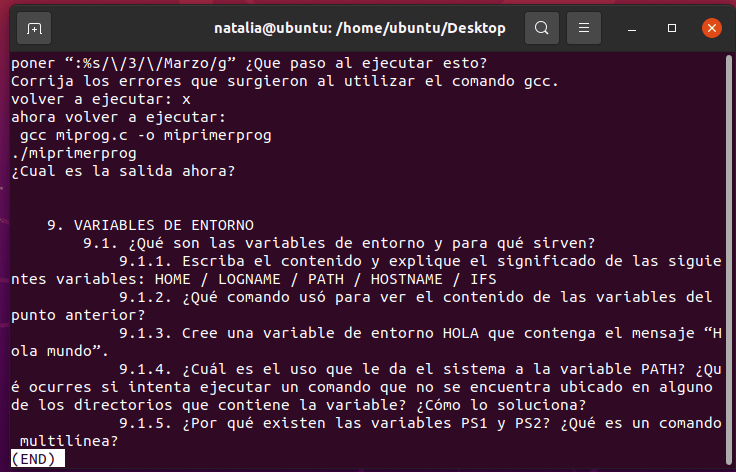
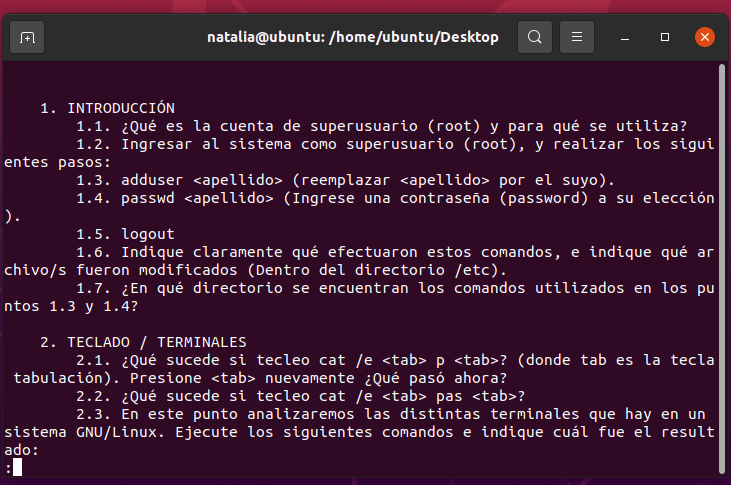
**Imagen 1**. Ejemplo de comando **cat prueba.txt**



**Imagen 2**. Ejemplo de comando **more prueba.txt**

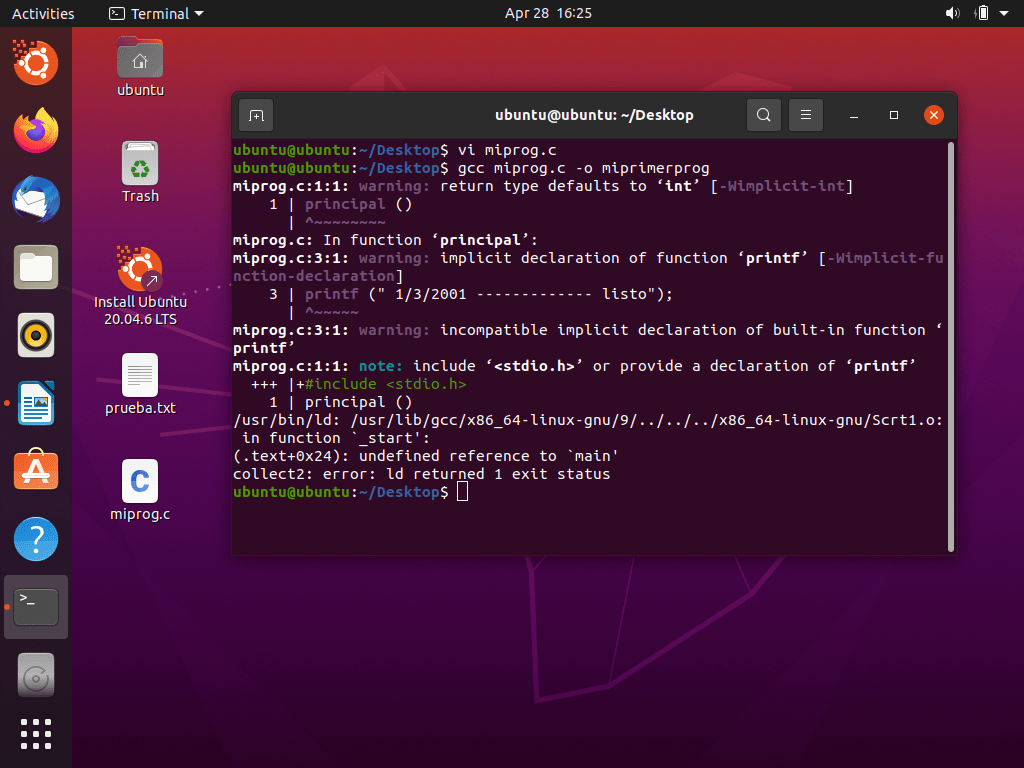


**Imagen 3.** Ejemplo de comando less **prueba.txt**

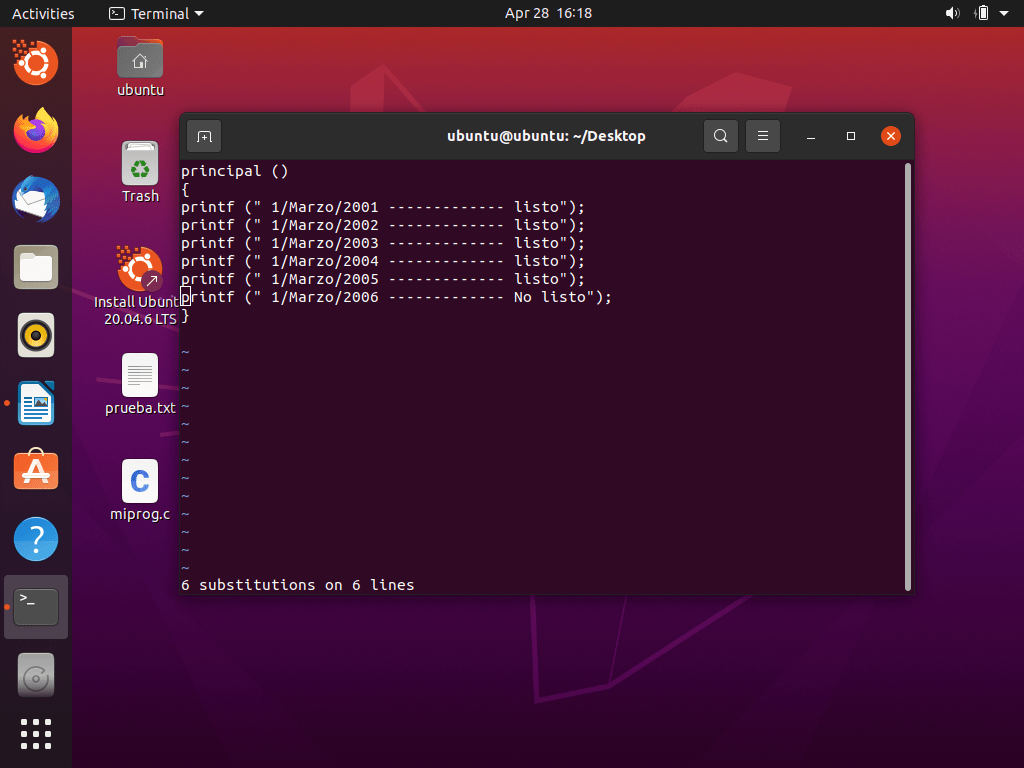
****

**PUNTO 8.1.17**

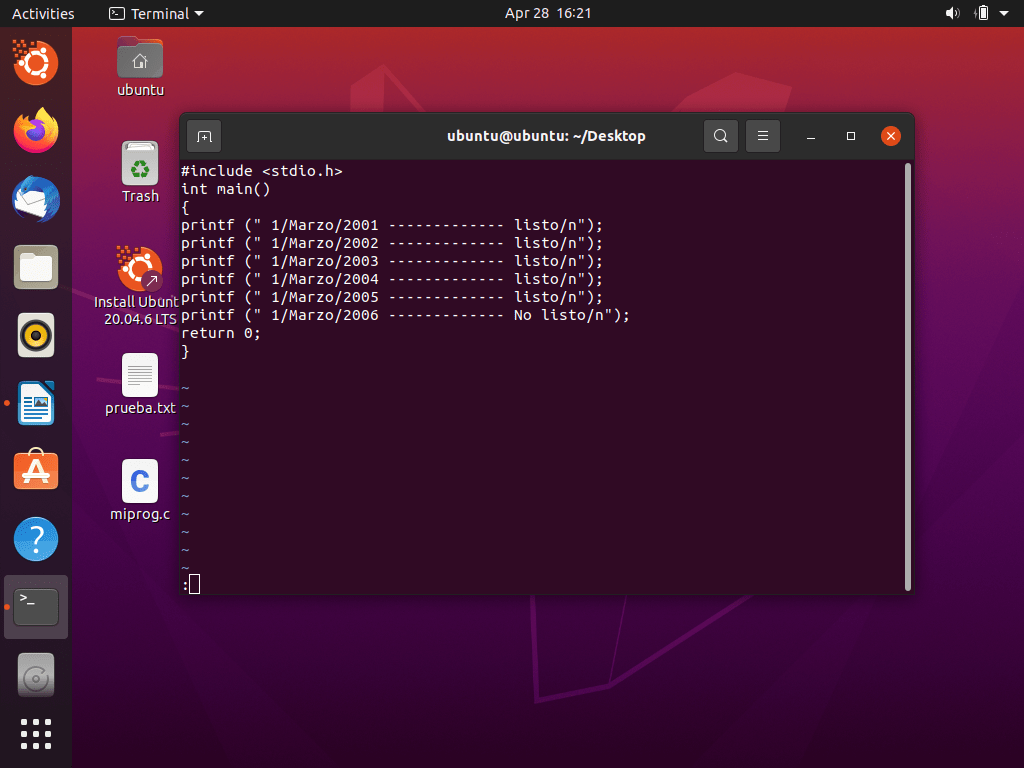
**Imagen 4.** Salida de pantalla del error al ejecutar por primera vez el comando gcc miprog.c -o miprimerprog ./miprimerprog.



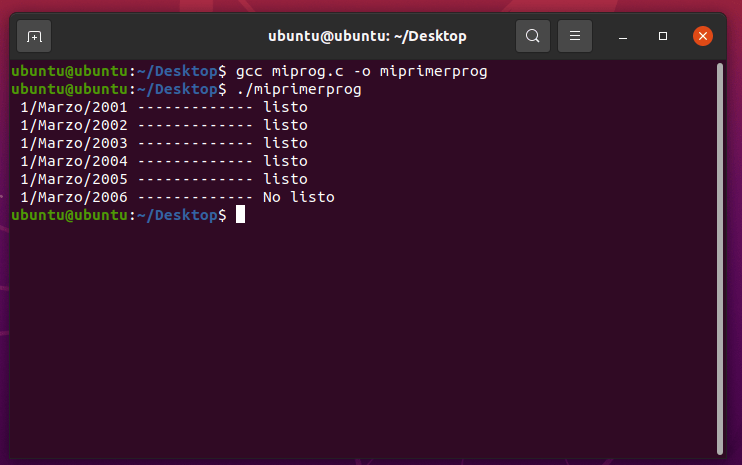
**Imagen 5.** Ejecución del comando “:%s/\/3/\/Marzo/g”



**Imagen 6.** Errores corregidos.



**Imagen 7** Salida por pantalla con errores corregidos.



**PUNTO 9**

**Imagen 8.** Contenido de las variables de entorno

