

# Fundamentos de Linux

Código de Curso: CY310

Versión: 3.0

# Prerrequisitos

---

- Ninguno

# Descripción del Curso

---

- Este curso está diseñado para iniciar a los estudiantes en el aprendizaje y manejo del Sistema Operativo Linux
- El alcance del curso es proporcionar los conceptos básicos relacionados:
  - ✓ Estructura de archivos
  - ✓ Shell (Intérprete de comandos)
  - ✓ Procesos
  - ✓ Personalización del ambiente en Linux

# Objetivos del Curso

---

- Establecer qué es un sistema operativo
- Explicar la historia de Linux y sus requerimientos de hardware
- Describir la organización del sistema operativo Linux
- Establecer qué es:
  - un Kernel (núcleo),
  - un sistema de archivos y
  - un Shell (intérprete de comandos) en Linux
- Entender el procesamiento de texto y la programación en sistemas operativos Linux
- Realizar las operaciones de ingreso y salida del sistema
- Explicar el formato de los comandos en Linux

# Objetivos del Curso... 2

---

- Entender cómo usar ciertos comandos básicos
- Explicar el uso de tuberías y filtros
- Explicar cómo Linux organiza su estructura de archivos
- Explicar el uso del editor vi
- Proporcionar una visión general del Shell (intérprete de comandos) de Linux
- Explicar las variables predefinidas del Shell
- Explicar los parámetros posicionales
- Personalizar las variables de entorno en una instalación Linux

# Agenda

---

Cada unidad en este curso es de dos horas de duración

## **Fundamentos de Linux y Sistemas de Archivos**

## Fundamentos de Linux



# Objetivos de Aprendizaje

---

- Explicar qué es un sistema operativo
- Explicar las características del sistema operativo UNIX
- Discutir la evolución de Linux
- Explicar los requerimientos de hardware de Linux
- Describir la organización del sistema operativo Linux
- Discutir acerca del kernel (núcleo), el shell (intérprete de comandos) y el sistema de archivos en Linux
- Discutir la capacidad de Linux para procesar texto, programas y proporcionar documentación para comandos

# Introducción

---

- Una computadora es capaz de realizar diferentes tareas:
  - Ejecutar programas de usuario
  - Conectar computadoras en una Red de Área Local (Local Area Network - LAN)
  - Compartir recursos
  - Controlar el hardware
- Para realizar estas tareas la computadora requiere de un programa especial denominado *sistema operativo*.

# Sistema Operativo

---

- El sistema operativo:
  - Reside en el disco duro de la computadora
  - Actúa como un puente entre los programas de usuario y los programas que controlan el hardware de la computadora
  - Controla todo el trabajo de la computadora
- El sistema operativo es responsable de las siguientes funciones del sistema de computadora:
  - Arrancar o iniciar la computadora
  - Actuar como interfaz entre el CPU y el mundo exterior
  - Coordinar los dispositivos del sistema
  - Coordinar las aplicaciones o programas en ejecución



# Rol de un Sistema Operativo

---



# Usos de un Sistema Operativo

---

- Proporciona diferentes recursos a los usuarios tales como cálculo, almacenamiento, dispositivos de Entrada/Salida, manejo de red, etc.
- Permite que varios usuarios trabajen juntos compartiendo e intercambiando programas, aplicaciones y datos en la misma instalación.
- Ayuda a resolver conflictos cuando los usuarios solicitan el mismo recurso simultáneamente.
- Proporciona seguridad cuando los usuarios comparten datos y programas.
- Asiste en la administración, y evaluación del uso y eficiencia de un sistema, recolectando datos sobre la utilización de recursos.

# Inicio de un Sistema Operativo

---

- El proceso de iniciar el sistema operativo se denomina *arranque (bootstrapping o booting)*
- Las instrucciones para el arranque están incluidas en uno de los chips de la computadora, el chip **BIOS** (Basic Input/Output System)
- El chip BIOS informa a la computadora que busque un programa especial llamado el *gestor de arranque (boot loader)*
- El gestor de arranque está disponible en un lugar fijo en el *disco de arranque*.
- El disco de arranque en cualquier computadora es el disco duro primario
- El gestor de arranque inicia la parte principal del sistema operativo



# Sistema Operativo UNIX

---

- UNIX es un sistema operativo 'abierto' porque puede ser llevado e instalado en cualquier clase de sistema de computadora y plataforma de hardware
- UNIX está escrito en un lenguaje de alto nivel y su código fuente está disponible fácilmente
- Es un sistema multiusuario y multitarea
- Una de sus características principales es compartir recursos en tiempo real por lo que es uno de los sistemas operativos más poderosos que existe
- Proporciona un entorno tan flexible que se usa en negocios, ciencias, educación, e industria
- Uno de de los usos más importantes del sistema operativo UNIX es en los interruptores de telecomunicación y los sistemas de transmisión



# Sistema Operativo Linux

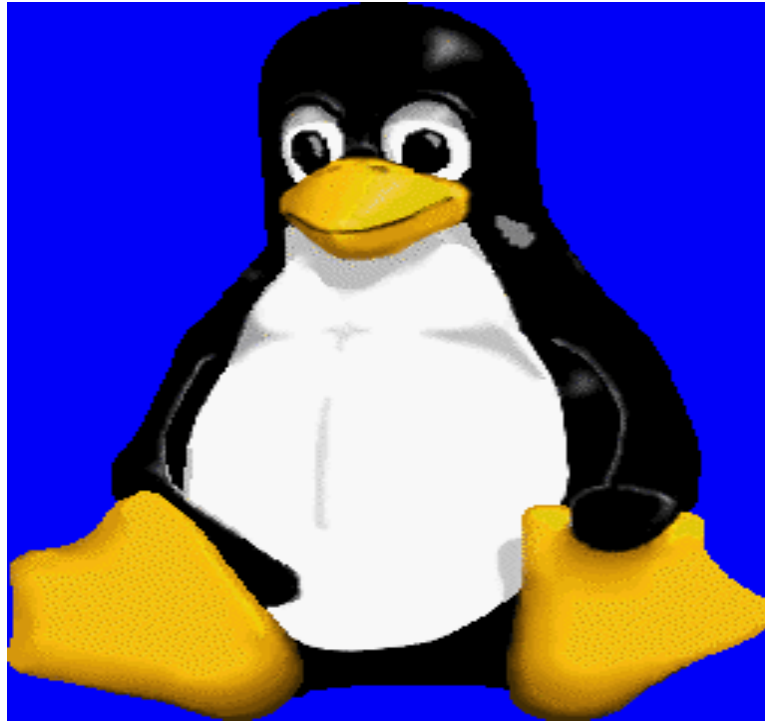
---

- Linux fue desarrollado, escrito, distribuido, y cubierto bajo Licencia Pública General de GNU (GPL)
- Su código fuente puede ser distribuido gratuitamente y está disponible para el público en general
- Los sistemas Linux se usan para el trabajo de redes, el desarrollo de software, alojar soluciones basadas en web y como plataforma de usuario final
- Linux es un sistema operativo estable y versátil, especialmente como un servidor de red
- Proporciona un sólido entorno gráfico, paquetes fáciles de instalar y aplicaciones de alto nivel



# Mascota de Linux

---



Tux

# Requerimientos de Hardware de Linux

| Hardware                     | Requerimientos   |
|------------------------------|--|
| CPU                          | La serie x86 de Intel y sus compatibles, DEC Alpha, Motorola Power PC, etc.  |
| Tarjeta Madre                | Sistemas de bus PCI, EISA, VESA y MCA.   |
| Memoria                      | 4 MB (mínimo), 16 MB recomendados para mayor eficiencia en ejecución.  |
| Monitor y Adaptador de Video | CGA, EGA, VGA, IBM monochrome, Super VGA, y otras tarjetas aceleradoras de vídeo   |
| Dispositivos de Puntero      | Mouse serial estándar Logitech, serie MM, Microsoft 2-botones, sistemas Mouse de 3-botones, etc  |
| Controlador de Disco Duro    | IDE, EIDE, MFM \,RLL y mayoría de controladores ESDI.  |
| Espacio de Disco Duro        | La instalación mínima requiere mínimo 100 MB de espacio. Una instalación completa de todos los servicios, los requerimientos pueden ser hasta de 2 GB. |



# Distribuciones de Linux

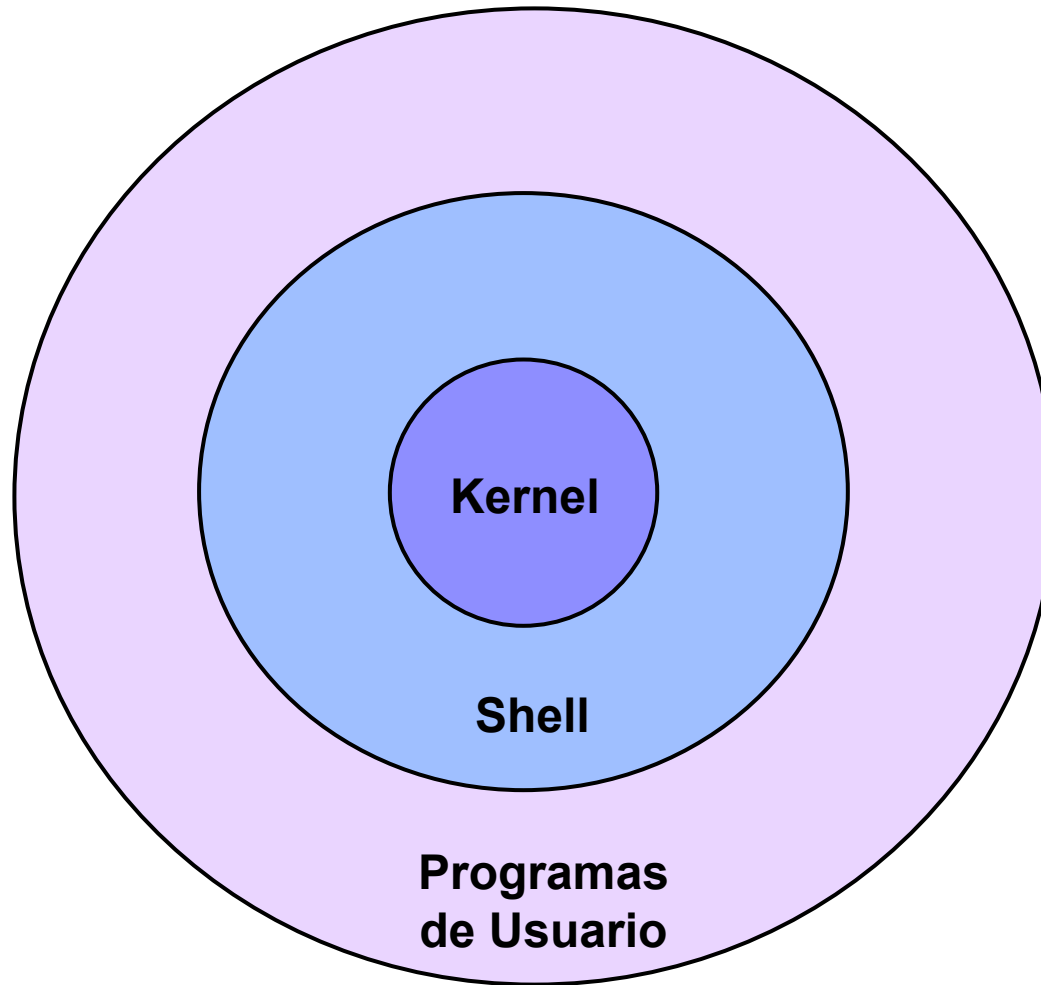
---

- ✓ Mandrake Linux desarrollado por MandrakeSoft
- ✓ Red Hat Linux desarrollado por Red Hat
- ✓ Debian GNU/Linux desarrollado por Debian
- ✓ SuSE Linux desarrollado por SuSE, Inc.
- ✓ Gentoo Linux desarrollado por Gentoo Technologies, Inc.
- ✓ El Proyecto Slackware Linux desarrollado por Slackware Linux, Inc.
- ✓ Lycoris Desktop/LX desarrollado por Lycoris
- ✓ Beehive Linux desarrollado por el Equipo Beehive
- ✓ Caldera OpenLinux desarrollado por Caldera International.
- ✓ Turbolinux desarrollado por Turbolinux, Inc



# Organización de Linux

---



# El Kernel (Núcleo)

---

## **El kernel realiza las siguientes tareas:**

- Verificar si el usuario es un usuario autorizado
- Guardar registro de los diferentes programas que están siendo ejecutados y asignar un tiempo específico a cada programa
- Asignar espacio de almacenamiento para archivos en el sistema
- Ejecutar el programa shell (Intérprete de comandos)
- Manejar la transferencia de información entre la computadora y las terminales

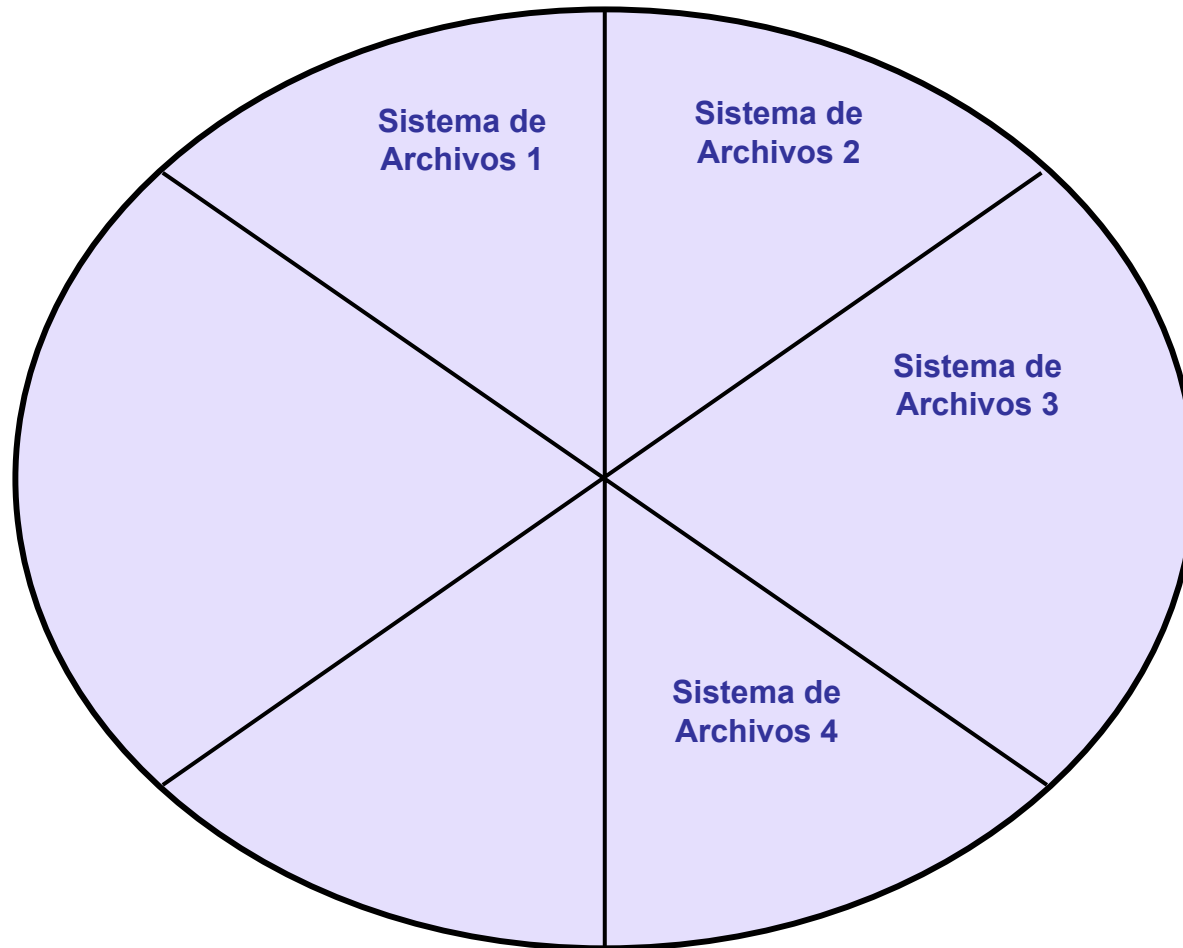
# El Shell (Intérprete de Comandos)

---

- El sistema operativo Linux usa un shell para transferir los comandos desde el teclado a la computadora
- El shell es el programa que toma comandos y ejecuta el programa apropiado o los traduce en instrucciones que el kernel entiende
- El shell es una interfaz basada en texto para el sistema operativo Linux
- El shell por defecto en Linux es bash (Bourne Again SHell)

# Sistema de Archivos en Linux

---



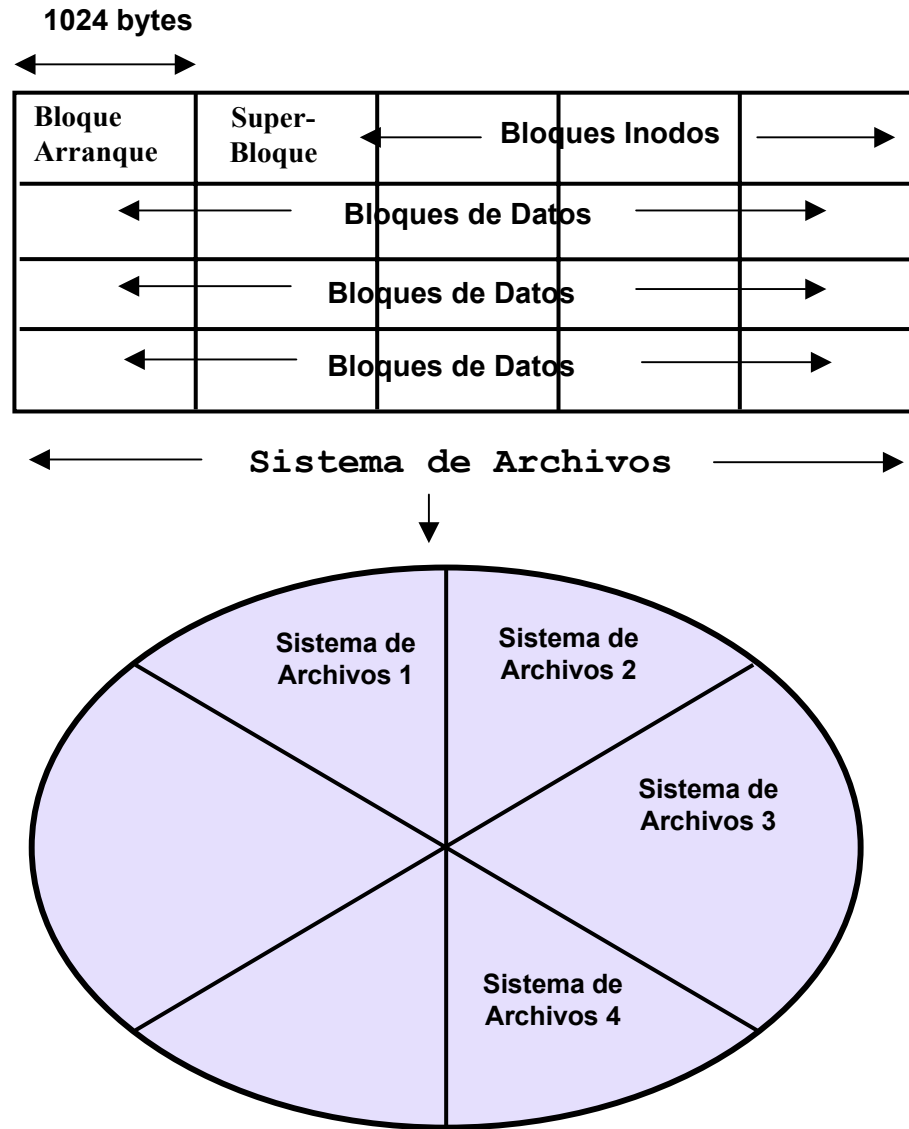
# Sistema de Archivos... 2

---

- El área de la superficie, donde se almacenan los archivos, está dividida en pistas circulares
- Las pistas circulares están divididas en *sectores* o *bloques* de disco
- Todos los bloques del disco son del mismo tamaño y tienen un número único llamado *número de bloque de disco*
- El tamaño del bloque de disco varía dependiendo de la distribución de Linux
- Los bloques de disco están organizados en los siguientes cuatro grupos:
  - Bloque de Arranque (Boot)
  - Súper bloque
  - Bloque inodo
  - Bloque de datos



# Unidad Típica del Sistema de Archivos



# Bloque de Arranque (Boot Block)

---

- El Bloque de Arranque (Boot Block) consiste de un bloque de disco que contiene el código para iniciar la computadora
- Ocupa el primer bloque de un sistema de archivos
- Un sistema sólo requiere de un bloque de arranque para iniciar el sistema
- En el resto de los sistemas de archivos, este bloque permanece vacío

# Súper Bloque

---

- El Súper Bloque está a continuación del bloque de arranque en el sistema de archivos
- Consiste de un bloque de disco, que contiene información acerca del sistema de archivos
- Contiene información acerca de:
  - El número de bloques en el sistema de archivos,
  - El número de bloques asignados para inodos y
  - El número de bloques que están actualmente libres

# Bloques Inodo y de Datos

---

- Los inodos son el tercer grupo de bloques en un sistema de archivos
- Contienen más de un bloque de disco para mantener información acerca de los archivos en el sistema de archivos
- El bloque de datos almacena el contenido del archivo
- El bloque de datos sigue a los bloques de disco asignados para inodos
- Un sistema de archivos contiene un cierto número de bloques de datos

# Partición de Disco

---

- Cada partición del disco consiste de bloques, situados en forma contigua, pero separados de las otras particiones
- La partición puede ser un sistema de archivos o un *espacio de intercambio* (*space swap*)
- Un espacio de intercambio se usa para implementar la memoria virtual, donde una porción de la memoria principal se almacena temporalmente
- La partición primaria es donde se almacenan los archivos relacionados al arranque
- Las particiones de espacio de intercambio son una secuencia lineal de bloques
- El tamaño de los archivos cambia a través del tiempo (crece o disminuye)

# Procesamiento de Texto

---

- El sistema Linux proporciona métodos poderosos de procesamiento de texto
- Para realizar procesamiento de texto están disponibles las herramientas:
  - **grep**
  - **egrep**
  - **Fgrep**
- También existen otras herramientas de procesamiento de texto, conocidas como editores.
- Los editores proporcionan las funcionalidades para crear, editar (modificar) y guardar texto.
- Algunos ejemplos de editores son:
  - **vi**
  - **ed**
  - **sed**
  - **emacs**

# Programación

---

- Se puede programar a través del shell, se conoce como *programación de shell*
- Linux proporciona más de un shell
- Cada shell en Linux proporciona la capacidad de programación
- El shell que se usa comúnmente es el Bourne Again Shell (popularmente conocido como *bash*)
- Un programa shell puede invocar las herramientas proporcionadas en Linux por medio de una sintaxis simple

# Documentación

---

- Linux proporciona una documentación elaborada para todas sus herramientas
- Las herramientas se les refiere comúnmente como comandos
- Algunos de los comandos son:
  - clear** - limpia la pantalla
  - date** - muestra la fecha y hora
  - cal** - muestra el calendario del mes actual
  - who** - muestra todos los usuarios que están actualmente conectados al sistema
- Algunos ejemplos de uso son:
  - man clear**
  - man date**
  - man man**



# Características del Sistema Linux

---

**El sistema Linux ofrece las siguientes características:**

- ✓ Estabilidad
- ✓ Multitarea
- ✓ Multiusuario
- ✓ Multiplataforma
- ✓ Gestión de Memoria
- ✓ Interfaz Gráfica de Usuario
- ✓ Desarrollo de Software
- ✓ Trabajo de redes
- ✓ Todo el código fuente está disponible

# Software disponible en Linux

---

**Varios tipos de software están disponibles en Linux:**

- ✓ Aplicaciones
- ✓ Software de Desarrollo
- ✓ Software Científico
- ✓ Software de Sistema
- ✓ Utilitarios
- ✓ Juegos

# Resumen

---

- Se explicó qué es un sistema operativo
- Se explicaron las características de un sistema operativo UNIX
- Se discutió acerca de la evolución de Linux
- Se explicaron los requerimientos de hardware de Linux
- Se describió la organización del sistema operativo Linux
- Se explicaron los conceptos de:
  - kernel (núcleo),
  - shell (intérprete de comandos),
  - sistema de archivos en Linux
- Se estudió la capacidad de Linux para procesar texto, programas y proporcionar documentación para comandos

## El Sistema Linux

# Objetivos de Aprendizaje

---

- Explicar el procedimiento de ingreso y salida en un sistema operativo Linux
- Discutir el uso del formato de comandos de Linux
- Describir la redirección de entrada y salida
- Explicar el uso de algunos comandos Linux básicos
- Discutir el uso de tuberías (pipes) y filtros

# Introducción

---

- Linux es un sistema operativo multiusuario y multitarea
- Varios usuarios pueden ejecutar varias aplicaciones en forma simultánea en una única computadora centralizada (procesador único)
- Linux cumple con POSIX (Portable Operating System Interface for UNIX)
- Es un sistema operativo estable y versátil

# Ingreso y Salida (Logging In and Out)

---

- Los usuarios en un sistema operativo multiusuario trabajan en terminales, esto comprende una unidad de salida, conocida como monitor (display), y un teclado
- Los terminales tienen números únicos asociados con ellos y están conectados a la unidad principal, que no necesariamente reside en la misma ubicación
- En un sistema que permite que varios usuarios operen en forma concurrente, debe existir una forma de identificar a los usuarios
- El sistema también debe asegurar que un usuario no suplante a otro y obtenga acceso a archivos confidenciales
- Todo esto se logra a través del procedimiento de registro (login)



# Procedimiento de Ingreso

---

- Cada usuario del sistema es identificado con una cuenta única
- Los usuarios son identificados por su nombre de registro (login name) y sus cuentas están protegidas a través de contraseñas de usuario (user passwords)
- Cada sistema Linux tiene un administrador del sistema que administra, monitorea y maneja el sistema
- Crear cuentas y proporcionar *nombres de registro* a los usuarios es una de las responsabilidades más importantes de un administrador de sistema



# Procedimiento de Ingreso... 2

---

- El administrador del sistema también proporciona la contraseña inicial para cada cuenta
- *root* es un usuario del sistema en el sistema Linux, que tiene el control completo del sistema operativo
- Al ingresar como *root*, el administrador del sistema puede realizar cualquier tarea en el sistema operativo
- Cada sistema tiene un *nombre de servidor (host name)* asignado, que facilita la rápida identificación de una máquina en una red

# Ingreso (Logging In)

---

- Una solicitud típica de ingreso y registro en un sistema Linux es:

**login:**

- El usuario tiene que ingresar el nombre de registro
- Ejemplo:
  - Se ingresa **mickeymouse**
  - Inmediatamente se muestra la solicitud de contraseña:  
**login: mickeymouse**  
**password:**
  - El usuario tiene que ingresar la contraseña
  - La contraseña debe ingresarse cuidadosamente dado que estos no aparecen en la pantalla
  - Si la contraseña no se ingresa correctamente, aparecerá el siguiente mensaje de error  
**Login incorrect**

# Salida (Login Out)

---

- Antes de terminar la sesión con el sistema, es importante que se salga (*log out*) del sistema
- Esto previene que otros usuarios puedan hacer mal uso de sus archivos
- Para salir del sistema se puede ingresar cualquiera de los siguientes comandos:

**logout**

**exit**

**Ctrl+D** (*Mantener presionado el botón Ctrl y  
presionar D en el teclado*)

# Formato de Comandos

---

- Todas las tareas en un sistema Linux pueden realizarse a través de comandos
- Una línea de comandos se puede estructurar en tres partes:
  - ✓ **El nombre del comando** es el nombre del comando que realiza una tarea específica
  - ✓ **Opciones:** Para la mayoría de comandos puede darse una o más opciones para que los resultados sean más específicos
  - ✓ **Argumentos:** Los argumentos para los comandos son usualmente archivos o directorios de donde los datos pueden ser leídos para que el comando opere sobre ellos
- La sintaxis de una línea de comandos es:

`comando -opción1 -opción2 ...-opciónn arg1 arg2 ... argm`

# Uso de Comandos

---

```
[mickeymouse@mycomputer mickeymouse]$ passwd  
Changing password for mickeymouse  
(current) UNIX password: mm123  
New UNIX password: jerry1960  
Retype new UNIX password: jerry1960  
passwd: all authentication tokens updated  
successfully  
[mickeymouse@mycomputer mickeymouse]$
```

# Uso de Comandos

---

```
[mickeymouse@mycomputer mickeymouse]$ man clear
```

```
clear(1)
```

```
clear(1)
```

```
NAME
```

**clear** – Limpia la pantalla del terminal

# Uso de Comandos

---

Sintaxis de cat: `cat [opciones] [archivo]`

```
[mickeymouse@mycomputer mickeymouse]$ cat myfile.txt
```

```
Este es un archivo de ejemplo.
```

```
Creado el 28 de Junio del 2001.
```

```
Hora de creación es 8:00 pm IST.
```

```
[mickeymouse@mycomputer mickeymouse]$
```

# Uso de Comandos

---

```
[mickeymouse@mycomputer mickeymouse]$ cat -n myfile.txt
1  Este es un archivo de ejemplo.
2  Creado el 28 de Junio del 2001.
3  Hora de creación es 8:00 pm IST.
[mickeymouse@mycomputer mickeymouse]$
```

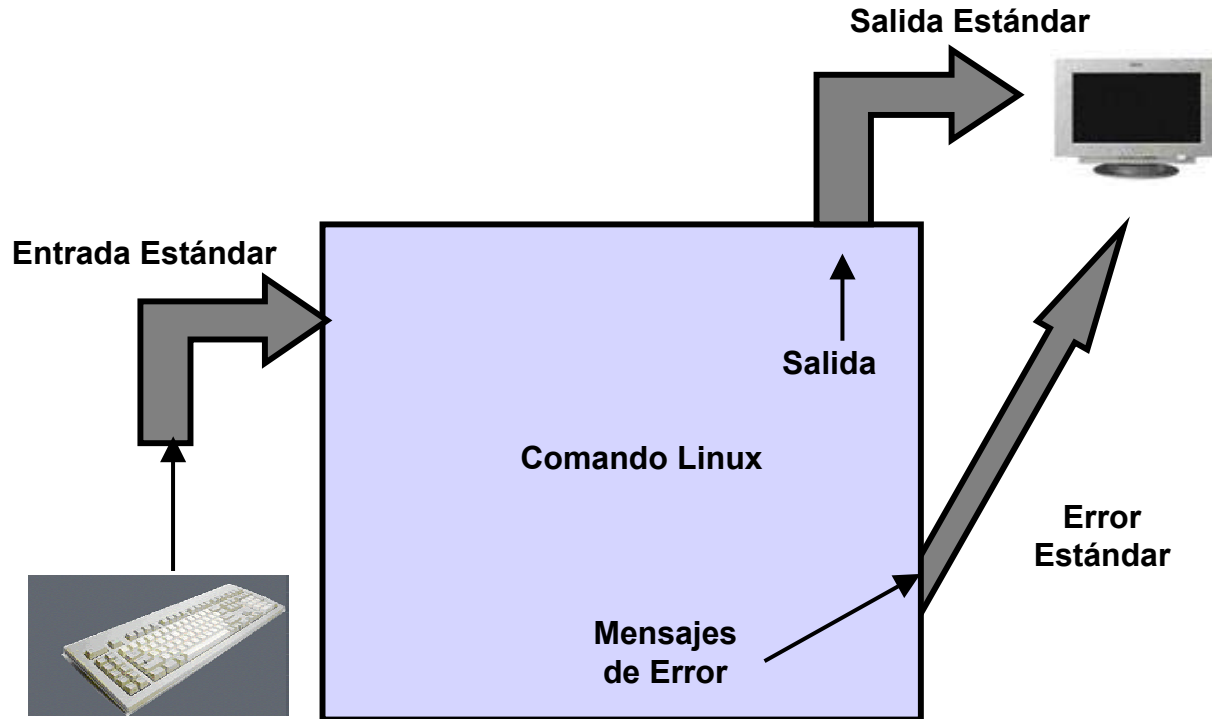


# Entrada, Salida y Error Estándar

---

- En todos los sistemas Linux, cualquier programa incluyendo todos los comandos Linux, está conectado automáticamente a tres archivos:
  - Entrada estándar
  - Salida estándar
  - Error estándar

# Entrada, Salida y Error Estándar



# Redirección de EntradaSalida (ES)

---

- Un programa puede tomar la entrada desde un archivo y enviar la salida hacia un archivo a través de la *redirección*
- La redirección es una manera a través de la cual se puede cambiar tanto la entrada estándar como la salida estándar
- Se usarán los operadores de redirección > y < para salida y entrada respectivamente
- El operador > hace que el archivo que sigue al operador la nueva salida estándar
- El operador < cambia la entrada estándar al archivo que sigue al operador
- El uso de los operadores > y < para redireccionar es temporal, cuando el comando finaliza se establecen nuevamente los valores por defecto

# Uso de Comandos

---

```
[mickeymouse@mycomputer mickeymouse]$ cat > newfile.txt
```

Este es el primer archivo que he creado.

Lo he creado el 29 de Junio del 2001.

ctrl-d

```
[mickeymouse@mycomputer mickeymouse]$
```

# Comandos Simples

---

- ✓ `cal`
- ✓ `date`
- ✓ `head`
- ✓ `tail`
- ✓ `sort`
- ✓ `cmp`
- ✓ `wc`
- ✓ `grep`
- ✓ `pr`
- ✓ `cut`

# Uso de Comandos

---

```
[mickeymouse@mycomputer mickeymouse]$ cal
```

July 2001

| Su | Mo | Tu | We | Th | Fr | Sa |
|----|----|----|----|----|----|----|
| 1  | 2  | 3  | 4  | 5  | 6  | 7  |
| 8  | 9  | 10 | 11 | 12 | 13 | 14 |
| 15 | 16 | 17 | 18 | 19 | 20 | 21 |
| 22 | 23 | 24 | 25 | 26 | 27 | 28 |
| 29 | 30 | 31 |    |    |    |    |

```
[mickeymouse@mycomputer mickeymouse]$
```

# Uso de Comandos

---

Sintaxis de date: `date [opcion] [+formato]`

```
[mickeymouse@mycomputer mickeymouse]$ date
```

```
Fri Apr 1 18:48:33 GMT 2005
```

```
[mickeymouse@mycomputer mickeymouse]$
```

*Nota : El comando date sin ningún argumento muestra el día, mes, fecha, hora y año actual*

```
[mickeymouse@mycomputer mickeymouse]$ date +%H
```

```
18
```

```
[mickeymouse@mycomputer mickeymouse]$
```

*date con el formato %H muestra solo la hora*

# Uso de Comandos

---

## Sintaxis de head: `head [opción] [archivo]`

```
[mickeymouse@mycomputer mickeymouse]$ head lines.txt
```

- *Muestra las primeras 10 líneas del archivo `lines.txt`.*

*El valor por defecto es 10*

```
[mickeymouse@mycomputer mickeymouse]$ head -4 lines.txt
```

```
Una mentira nunca vive para ser vieja. -- Sófocles
Un hombre honesto es siempre un niño. -- Martial
Cuando estés en duda, di la verdad. - Mark Twain
Las palabras falsas no son solo malas en sí mismas,
sino que infectan el alma con maldad. - Socrates
```

```
[mickeymouse@mycomputer mickeymouse]$
```

- *El comando anterior muestra las primeras 4 líneas del archivo*



# Uso de Comandos

---

Sintaxis de tail: `tail [opción] [archivo]`

```
[mickeymouse@mycomputer mickeymouse]$ tail lines.txt
```

- *El comando imprime las 10 últimas líneas*

```
[mickeymouse@mycomputer mickeymouse]$ tail -c10 lines.txt
```

*bio Arabe*

```
[mickeymouse@mycomputer mickeymouse]$
```

- *Muestra los 10 últimos bytes (caracteres) del archivo*

# Uso de Comandos

---

Sintaxis de sort: `sort [opción] [archivo]`

```
[mickeymouse@mycomputer mickeymouse]$ sort names.txt
```

asterix

donald

jerry

mickey

noddy

obelix

tintin

tom

```
[mickeymouse@mycomputer mickeymouse]$
```

- *Ordena las líneas del archivo **names.txt** y muestra los datos ordenados*

```
[mickeymouse@mycomputer mickeymouse]$ sort -r names.txt
```

- *Ordena los datos en orden inverso (descendente)*

# Uso de Comandos

---

Sintaxis de cmp: `cmp[opción]archivo1 archivo2`

*Las únicas dos opciones permitidas para `cmp` son `-l` y `-s`.*

```
[mickeymouse@mycomputer mickeymouse]$ cmp names.txt  
lines.txt  
names.txt lines.txt differ: char 1, line 1  
[mickeymouse@mycomputer mickeymouse]$
```

# Uso de Comandos

---

Sintaxis de wc: `wc [opciones] [archivo]`

```
[mickeymouse@mycomputer mickeymouse]$ wc names.txt
```

```
      8      8      52 names.txt
```

- *El comando muestra el número de líneas, palabras y caracteres (en ese orden) de **names.txt***

```
[mickeymouse@mycomputer mickeymouse]$ wc names.txt lines.txt
```

```
      8      8      52 names.txt
     11     142     707 lines.txt
     19     150     759 total
```

- *El comando muestra el número de líneas, palabras y caracteres de ambos archivos, uno a continuación del otro, y también el total*

# Uso de Comandos

---

Sintaxis de grep: `grep [patrón] [archivo]`

```
[mickeymouse@mycomputer mickeymouse]$ grep mentira lines.txt
Una mentira nunca vive para ser vieja. -- Sófocles
No me importa la mentira, pero odio la inexactitud. -- Samuel
Butler
Me hago a mí mismo un daño mayor mintiendo, que aquel que
ocasiono a quién le he dicho una mentira. -- Michel De
Montaigne
[mickeymouse@mycomputer mickeymouse]$
```

# Uso de Comandos

---

Sintaxis de pr: `pr [opción] [archivo]`

```
[mickeymouse@mycomputer mickeymouse]$ pr -2 names.txt
2001-06-22 17:58                      names.txt                      Page 1
noddy                                mickey
tom                                  asterix
jerry                                tintin
donald obelix
[mickeymouse@mycomputer mickeymouse]$
```

# Uso de Comandos

---

Sintaxis de cut: `cut [opción] [archivo]`

```
[mickeymouse@mycomputer mickeymouse]$ cut -f1 mylines.txt
```

```
Esta es mi primera línea de texto.
```

```
Esta es mi segunda línea de texto.
```

```
1 2 3 4 5 6 7
```

```
9 10 11 12 13 14
```

```
[mickeymouse@mycomputer mickeymouse]$
```

# Comandos Simples

---

- **talk login\_name**: El comando **talk** copia líneas de un usuario a otro cuando los usuarios están en línea
- **wall message**: Envía el **mensaje** ingresado por un usuario a todos los usuarios actualmente conectados
- **mail**: Permite a los usuarios tanto enviar como leer correos
- **who**: Muestra todos los usuarios que están conectados actualmente en el sistema
- **whoami**: Ayuda a encontrar quién está actualmente conectado



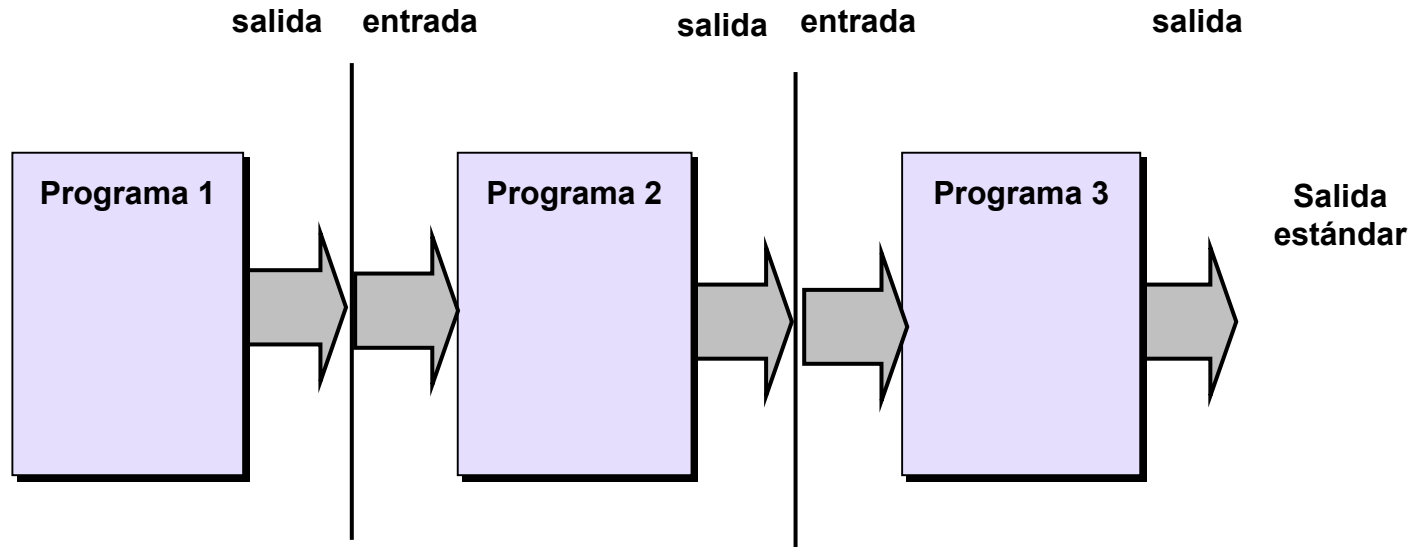
# Tuberías (Pipes)

---

- La redirección en Linux ayuda a conectar programas con archivos
- Las tuberías en Linux ayudan a conectar un programa con otros programas
- Las tuberías en línea permiten que la salida de un programa sea enviada como entrada a otro programa
- El carácter `|` (un carácter barra vertical) representa una tubería

# Tuberías en Linux

---



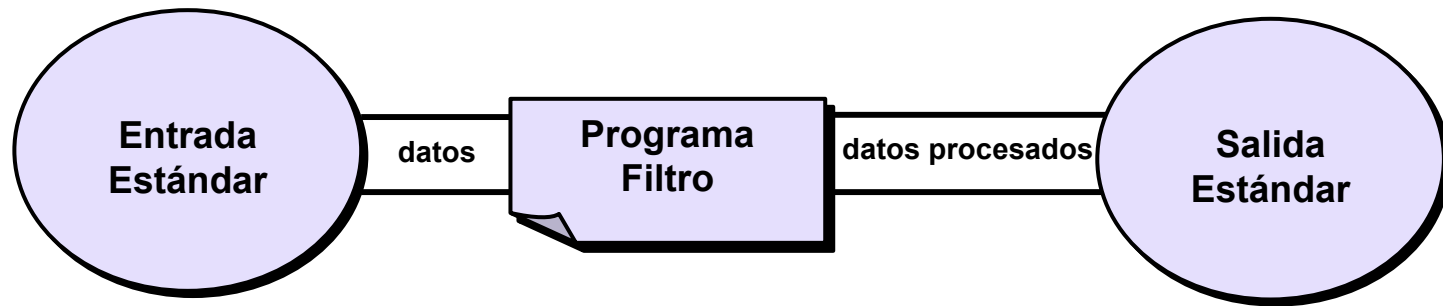
# Uso de Comandos

---

```
[mickeymouse@mycomputer mickeymouse]$  
cat lines.txt | grep hop | wc -l  
3  
[mickeymouse@mycomputer mickeymouse]$
```

# Filtros en Linux

---



# Resumen

---

- Se explicó el procedimiento de ingreso y salida en un sistema operativo Linux
- Se discutió el uso del formato de comandos de Linux
- Se estudió la redirección de entrada y salida
- Se explicó el uso de algunos comandos básicos de Linux
- Se discutió el uso de tuberías (pipes) y filtros

## Laboratorio del Sistema Linux

## **La Estructura de Archivos de Linux**

# Objetivos de Aprendizaje

---

- Explicar cómo Linux organiza su estructura de archivos
- Describir el uso de los comandos de archivos
- Discutir los directorios y los comandos relacionados a directorios
- Explicar cómo se dan permisos a los archivos



# Introducción

---

- Existen tanto una vista física como una vista lógica del sistema de archivos
- Los archivos son almacenados físicamente en el disco dentro de un sistema de archivos
- Lógicamente, un sistema de archivos en Linux puede ser visto como una estructura jerárquica (árbol)

# Principales Características de Archivos

---

- Cada recurso en Linux es un archivo
- Un archivo es una secuencia de bytes
- Cada archivo tiene un nombre.
- El uso de extensión en un nombre de archivo es opcional
- Los archivos son almacenes para datos
- Un comando `file` puede determinar el tipo del archivo

# Características de Directorios

---

- Los directorios pueden guardar cualquier tipo de archivo
- Todos los directorios tienen nombres, pero no tienen extensiones
- Un directorio Linux es un archivo especial que mantiene una lista de todos los archivos almacenados en él
- Un archivo en un directorio puede ser otro directorio
- El directorio hijo es llamado el *subdirectorio* del directorio original

# Componentes de Directorio

---

- ✓ Nombre
- ✓ Contenido
- ✓ Permisos
- ✓ Fecha y hora de Modificación

# Inodos

---

- La información acerca de los permisos y fechas de modificación para un archivo se denomina información administrativa
- La información administrativa se almacena en una tabla llamada i-nodo o inodo
- Cada archivo en Linux está asociado con un inodo

# Tipos de Tiempos de Modificación

---

**En los inodos se almacenan tres tipos de fecha de modificación:**

- La fecha y hora en que el contenido del archivo fue modificado por última vez
- La fecha y hora en que el archivo fue usado por última vez, sin causar ninguna modificación al archivo
- La fecha y hora en que el inodo fue cambiado, lo que ocurre cuando los permisos para el archivo son alterados

# Información almacenada en los Inodos

---

## Los Inodos contienen la siguiente información:

- **Identificación del propietario del archivo:** Los detalles respecto a los usuarios que tienen derechos de acceso al archivo
- **Tipo del archivo:** Información acerca del tipo de archivo: regular, directorio, carácter o de bloque especial
- **Número de enlaces al archivo:** Información acerca del número de nombres que tiene el archivo en la jerarquía del directorio
- **Tamaño del archivo:** Información acerca del tamaño del archivo en bytes
- **Lista de direcciones de disco para el archivo:** Detalles de la lista de las direcciones en el disco donde reside cada parte del archivo

# Ejemplo de un Inodo

---

|   |
|---|
| <b>Propietario/Grupo</b>                                |
| <b>Tipo de archivo</b>                                  |
| <b>Tamaño del archivo</b>                               |
| <b>Permisos del archivo (rwxrwxrwx)</b>                 |
| <b>Fecha/Hora de:<br/>Modificación, Acceso y Cambio</b> |
| <b>Cantidad de enlaces</b>                              |
| <b>Banderas adicionales (ACL,<br/>EXT2_FLAGS)</b>       |
| <b>Punteros a bloques de datos</b>                      |



# Comandos de Directorio

---

- Los comandos permiten:
  - ✓ crear un directorio
  - ✓ mover archivos a un directorio
  - ✓ eliminar directorios
  - ✓ encontrar la ruta del directorio de trabajo actual
  - ✓ listar su contenido
- Se discutirán los siguientes comandos:
  - ✓ **mkdir**
  - ✓ **cd**
  - ✓ **rmdir**
  - ✓ **pwd**
  - ✓ **ls**

# El Comando mkdir

---

El comando `mkdir` se usa para crear un directorio.

**Sintaxis de mkdir:** `mkdir [opción] directorio`

```
[mickeymouse@mycomputer mickeymouse]$ mkdir programas
```

- *Crea un directorio llamado programas en el directorio home del usuario mickeymouse*

```
[mickeymouse@mycomputer mickeymouse]$ mkdir documentos  
proyectos
```

- *Crea dos directorios más llamados documentos y proyectos en el mismo directorio home*

# El Comando mkdir... 2

---

- Cuando el administrador del sistema crea una cuenta para `mickeymouse`, se crea automáticamente en el directorio home del sistema, un directorio llamado `mickeymouse`
- El directorio home es un subdirectorio del directorio raíz (/)
- Los directorios `programas`, `documentos` y `proyectos` son subdirectorios del directorio `mickeymouse`.
- El comando `mkdir` puede tomar uno o más parámetros
- El comando `mkdir` crea un directorio con el nombre especificado si el directorio aún no existe.
- Si se trata de crear un directorio que ya existe, se muestra el siguiente mensaje de error:

```
[mickeymouse@mycomputer mickeymouse]$ mkdir programas  
mkdir: Cannot create directory 'programas' : File exists  
[mickeymouse@mycomputer mickeymouse]$
```

# El Comando cd

---

**cd** siglas de *change directory* (cambiar directorio)

## Sintaxis de cd: **cd directorio**

```
[mickeymouse@mycomputer mickeymouse]$ cd programas  
[mickeymouse@mycomputer programas]$
```

- El usuario *mickeymouse* ahora está en el directorio *programas*

Considere el nuevo prompt:

```
[mickeymouse@mycomputer programas]$
```

- Se muestra que el directorio en que se está es de hecho donde se quiere estar
- Existen dos símbolos especiales:
  - ✓ El “.” representa el directorio actual
  - ✓ El “..” representa el directorio padre

# El Comando `rmdir`

---

El comando `rmdir` se usa para eliminar un directorio existente del sistema de archivos.

**Sintaxis de `rmdir`:** `rmdir [opción] directorio`

```
[mickeymouse@mycomputer mickeymouse]$ rmdir proyectos
```

```
[mickeymouse@mycomputer mickeymouse]$
```

- Se intenta eliminar el directorio `proyectos`:
  - Si el directorio `proyectos` está vacío, simplemente lo elimina.
  - Caso contrario se muestra el siguiente mensaje:

```
rmdir: proyectos: Directory not empty
```

# El Comando pwd

---

El comando **pwd** (**print working directory**) no es un comando relacionado a directorios, pero proporciona información acerca del directorio de trabajo en que están los usuarios.

**Sintaxis de pwd:**      **pwd**    [opción]

```
[mickeymouse@mycomputer mickeymouse]$ pwd  
/home/mickeymouse
```

```
[mickeymouse@mycomputer mickeymouse]$ cd documentos  
[mickeymouse@mycomputer documentos]$ pwd  
/home/mickeymouse/documentos
```

- El nombre completo del directorio de trabajo actual **mickeymouse** es **/home/mickeymouse**.
- Cuando se cambia el directorio a **documentos**, el nombre completo es ahora **/home/mickeymouse/documentos**

# El Comando ls

---

El comando **ls** se usa para ver el contenido de un directorio

**Sintaxis de ls:**     **ls [opción] [archivo]**

```
[mickeymouse@mycomputer programas]$ ls
```

```
comentarios.txt cprograms como.txt programas
```

# El Comando ls... 2

---

- Esta es la forma más simple de usar el comando `ls`
- El contenido del directorio `programas` se lista a continuación:

```
[mickeymouse@mycomputer programas]$ ls - l
total 16
-rw-r--r--      1 mickeymouse 3012 Jun 30      10:30
comentarios.txt
drw-r--r--      1 mickeymouse 4096 Jun 30      11:35
cprograms
-rw-r--r--      1 mickeymouse  52   Jun 30      12:36
como.txt
-rw-r--r--      1 mickeymouse 158   Jun 30      13:49
programas
[mickeymouse@mycomputer programas]$
```



# Usos Válidos de Comandos

---

Los usos válidos de los comandos relacionados a directorios son:

```
/home/mickeymouse$ cd /home/jerry
```

- Es válido si **mickeymouse** tiene permiso para cambiar al directorio **jerry**

```
[mickeymouse@mycomputer mickeymouse]$ mkdir  
documentos/docespeciales
```

- Se crea un subdirectorio llamado **docespeciales** en el subdirectorio **documentos**

```
[mickeymouse@mycomputer mickeymouse]$ rmdir  
documentos/docespeciales
```

- Se elimina el subdirectorio **docespeciales**, si está vacío

```
[mickeymouse@mycomputer mickeymouse]$ ls  
/home/mickeymouse/documentos
```

# Comandos de Archivos

---

- Los archivos también se pueden crear con editores.
- Un editor es un programa que permite la creación y modificación de un archivo
- Un comando para crear archivos sólo permite crear archivos al usuario.
- Un editor, en cambio, permite al usuario crear y modificar el contenido del archivo

# Uso de Comando en el Editor ed

---

## Sintaxis de ed: ed [archivo]

```
[mickeymouse@mycomputer mickeymouse]$ ed
```

```
a
```

```
Este archivo fue creado usando el editor ed.  
Creado el 30 Junio 2001.
```

```
A las 11:30 AM.
```

```
.
```

```
w miarchivo.txt
```

```
86
```

```
q
```

```
[mickeymouse@mycomputer mickeymouse]$
```

# Uso de Comando en el Editor ed... 2

---

- Si se ejecuta el editor sin un nombre de archivo, se asume que es un nuevo archivo.
- El nombre de archivo puede ser especificado dentro del editor cuando se graben las líneas ingresadas.
- Para ingresar líneas existen se pueden usar dos comandos en **ed**:
  - El comando “**a**” de append (añadir)
  - El comando “**i**” de insert (insertar)
- En el ejemplo: Después del comando **a** se ingresaron tres líneas.
- Los comandos del editor **ed** son simples y deben estar en una línea ellos solos

# Uso de Comando en el Editor ed... 3

---

- Para indicar la finalización del ingreso, se ingresa el comando “.”
- El comando “.” informa al editor **ed** que está en modo de comando y no en modo de edición. El editor **ed** inicia el modo de comando.
- Las líneas que se ingresaron sólo están en la memoria principal de la computadora.
- Para escribir las líneas en un archivo en el disco, se usa el comando **w** seguido por el nombre del archivo

# Uso de Comando en el Editor ed... 4

---

- El nombre del archivo se ingresa sólo la primera vez.
- Posteriormente se puede ingresar simplemente el comando **w** y las líneas se guardarán en el mismo archivo
- El comando **w** escribe las líneas a un archivo en el disco, en este caso al archivo llamado **miarchivo.txt** y muestra el número de caracteres escritos, que en este caso es 86.
- El comando **q** termina el programa.
- Si se usa **q** sin haber escrito las líneas en el archivo, **ed** mostrará **?**

# Uso de Comando en el Editor ed... 5

---

```
[mickeymouse@mycomputer mickeymouse]$ cat miarchivo.txt
Este archivo fue creado usando el editor ed.
Creado el 30 Junio 2001.
A las 11:30 AM.
```

```
[mickeymouse@mycomputer mickeymouse]$
```

- Usando `cat` se puede ver que las tres líneas ingresadas a través de `ed` están correctas.

```
[mickeymouse@mycomputer mickeymouse]$ ed miarchivo.txt
86
a
Añadiendo otra línea.
Esta línea fue añadida 11:35 AM.
```

```
.
w
141
q
[mickeymouse@mycomputer mickeymouse]$
```



# Uso de Comando en el Editor ed... 6

---

- En el ejemplo se usa **ed** con el nombre del archivo a continuación.
  - Se abre el archivo si éste existe, en caso contrario muestra un mensaje de error y espera por otra entrada del usuario.
  - Si el archivo existe, se muestra el número de caracteres que contiene y se espera por la entrada del usuario.
  - En el ejemplo se han agregado dos líneas más al archivo **miarchivo.txt**. A continuación se muestra el archivo con las líneas añadidas usando el comando **cat**:

```
[mickeymouse@mycomputer mickeymouse]$ cat miarchivo.txt
Este archivo fue creado usando el editor ed.
Creado el 30 Junio 2001.
A las 11:30 AM.
Añadiendo otra línea.
Esta línea fue añadida 11:35 AM.
[mickeymouse@mycomputer mickeymouse]$
```





# Comandos Relacionados a Archivos

---

## El comando cp:

**cp** copia el contenido de un archivo a otro archivo.

## Sintaxis de cp: **cp[opción]origen destino**

```
[mickeymouse@mycomputer mickeymouse]$ cp miarchivo.txt  
nuevoarchivo.txt
```

El contenido de **miarchivo.txt** se copia a **nuevoarchivo.txt**.

El comando **cp** sobrescribe el archivo destino.

# Comandos Relacionados a Archivos... 2

---

## El comando mv:

Se usa para renombrar un archivo ya existente

Sintaxis de mv: `mv [opción] origen destino`

```
[mickeymouse@mycomputer mickeymouse]$ mv  
miarchivo.txt nuevoarchivo.txt
```

El comando dado cambia el nombre de `miarchivo.txt`  
a `nuevoarchivo.txt`

# Comandos Relacionados a Archivos... 3

---

El comando **rm** se usa para eliminar un archivo

**Sintaxis de rm:** **rm [opción] archivo**

```
[mickeymouse@mycomputer mickeymouse]$ rm miarchivo.txt
```

- Se elimina **miarchivo.txt** del directorio **mickeymouse**.
- Se puede eliminar archivos desde cualquier parte del sistema siempre que se tenga los permisos necesarios.
- Si hay más de un enlace al archivo, **rm** sólo elimina el enlace actual.
- El archivo se elimina completamente del sistema cuando todos los enlaces son eliminados.

# Comandos Relacionados a Archivos... 4

---

- El comando **rm -r** se puede usar para eliminar el contenido de los subdirectorios recursivamente
- Los usuarios deben ser muy cuidadosos al usar el comando con **-r**.
- El comando **rm -r** puede tomar un archivo como argumento.
- El comando con la opción **-r** opera de la siguiente forma:
  - ✓ Si el argumento es un archivo, entonces elimina el archivo.
  - ✓ Si el argumento es un directorio, entonces elimina el directorio y su contenido sin ningún aviso

# Permisos de Archivo

---

- Cada archivo en Linux tiene un conjunto de permisos asociados que ayudan a asegurar que los datos almacenados no sean dañados por otros.
- Los permisos le indican al sistema quién puede hacer qué con el archivo.
- Linux divide a los usuarios en tres categorías:
  - ✓ Usuario
  - ✓ Miembro de un grupo de usuarios
  - ✓ Otros usuarios

# Valores para Tipos de Permisos

---

- Linux asigna diferentes valores para estos tres tipos de permisos :
  - ✓ 4 para lectura
  - ✓ 2 para escritura
  - ✓ 1 para ejecución
- El sistema usa el *número octal* o el sistema *base-8* para establecer los permisos.
- Tiene 8 dígitos válidos, 0,1,2,3,4,5,6 y 7.
- Un número octal válido es una combinación de uno o más de cualquiera de los dígitos válidos del 0 al 7.

# El Comando chmod

---

Linux proporciona un comando **chmod** (del inglés change mode) para asignar permisos a los archivos y directorios y para cambiar los permisos ya dados

**Sintaxis de chmod:** `chmod [opción] modo archivo`

```
[mickeymouse@mycomputer mickeymouse]$
```

```
chmod 754 miarchivo.txt
```

```
[mickeymouse@mycomputer mickeymouse]$
```

- Este comando otorga permisos de lectura, escritura y ejecución para el usuario; lectura y ejecución para el grupo y solo lectura para los otros
- Recuerde que los archivos de texto sólo pueden ser leídos y modificados.
- Ejecutar un archivo involucra ejecutar un comando o un programa

# Resumen

---

- Se explicó cómo Linux organiza su estructura de archivos
- Se explicó el uso de los comandos de archivos
- Se discutió acerca de los directorios y los comandos relacionados a directorios
- Se explicó cómo se dan los permisos a los archivos



# Laboratorio

# Estructura de Archivos en Linux

## El Editor vi

# Objetivos de Aprendizaje

---

- Discutir cómo trabaja el editor visual vi en Linux
- Describir las diferentes categorías de comandos del editor vi
- Explicar el uso de comandos para editar archivos en vi

# Introducción

---

- Los editores se usan principalmente para crear documentos
- Linux ofrece varios editores: **vi**, **emacs**, **ed**, y **ex**
- Los editores **ed** y **ex** se denominan editores de 'línea'
- El editor **ed** permite realizar inserción, eliminación, modificación, etc., línea por línea
- El editor **vi** ofrece una facilidad de pantalla completa para crear y editar documentos

# Introducción al Editor vi

---

- El término '**vi**' viene de 'visual editor'
- **vi** es el único editor que se encuentra en casi todas las instalaciones de Unix/Linux
- **vi** proporciona un amplio conjunto de comandos para insertar, eliminar y modificar texto
- Se pueden realizar búsquedas y reemplazos de texto usando expresiones regulares
- Las expresiones regulares son fórmulas que hacen coincidir cadenas con un patrón dado

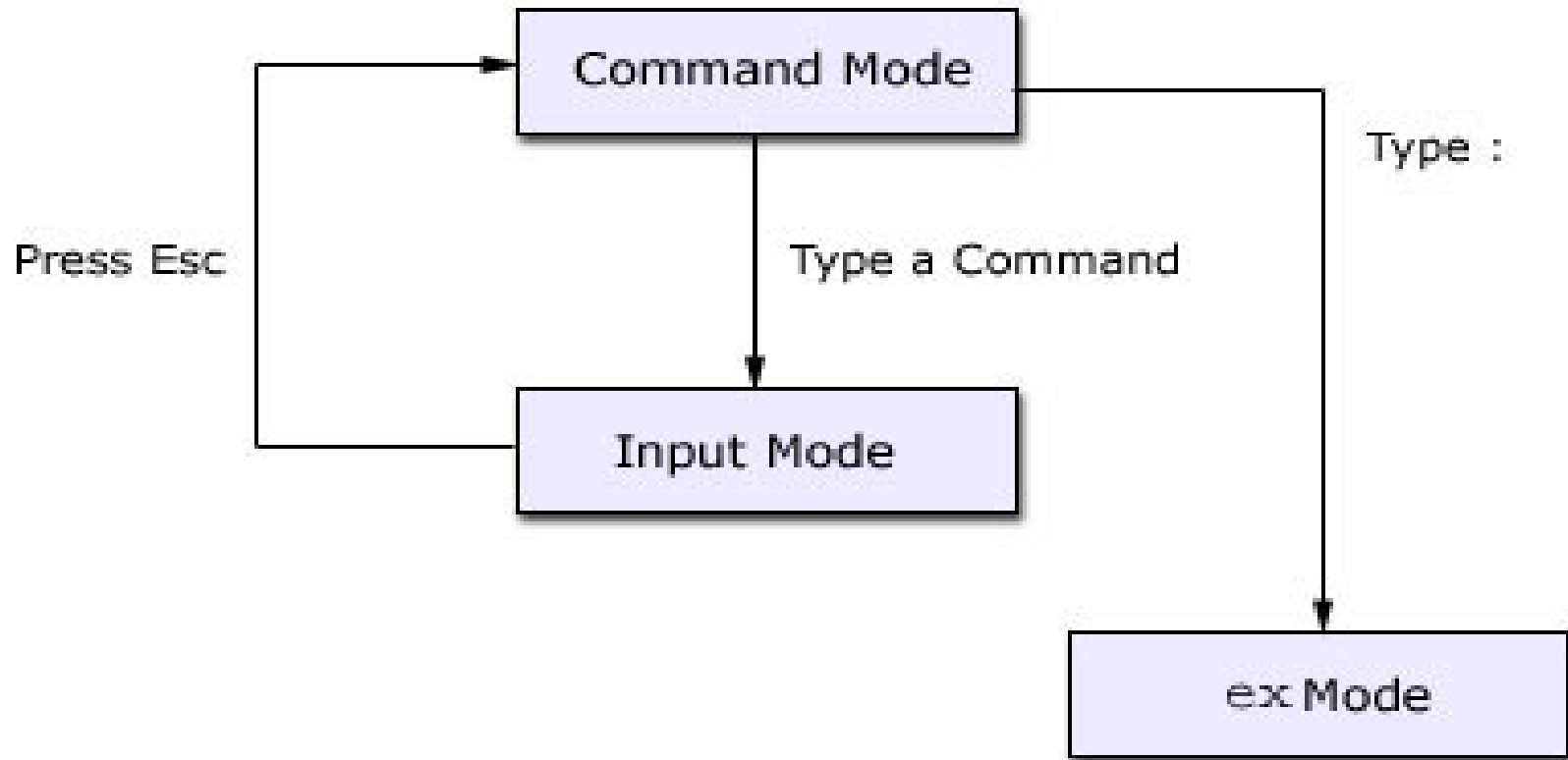
# Modos en vi

---

- El editor `vi` permite tres modos de edición:
  - **Comando**: Cuando se invoca el programa `vi`, está en el modo de comando donde sólo pueden usarse comandos válidos
  - **Entrada**: En este modo se puede ingresar, eliminar y modificar texto
  - **Modo `ex`**: En este modo se puede invocar cualquiera de los comandos del editor `ex`. Generalmente, la mayoría de los comandos para grabar usados en `vi` son comandos `ex`

# Modos de Edición en vi... 2

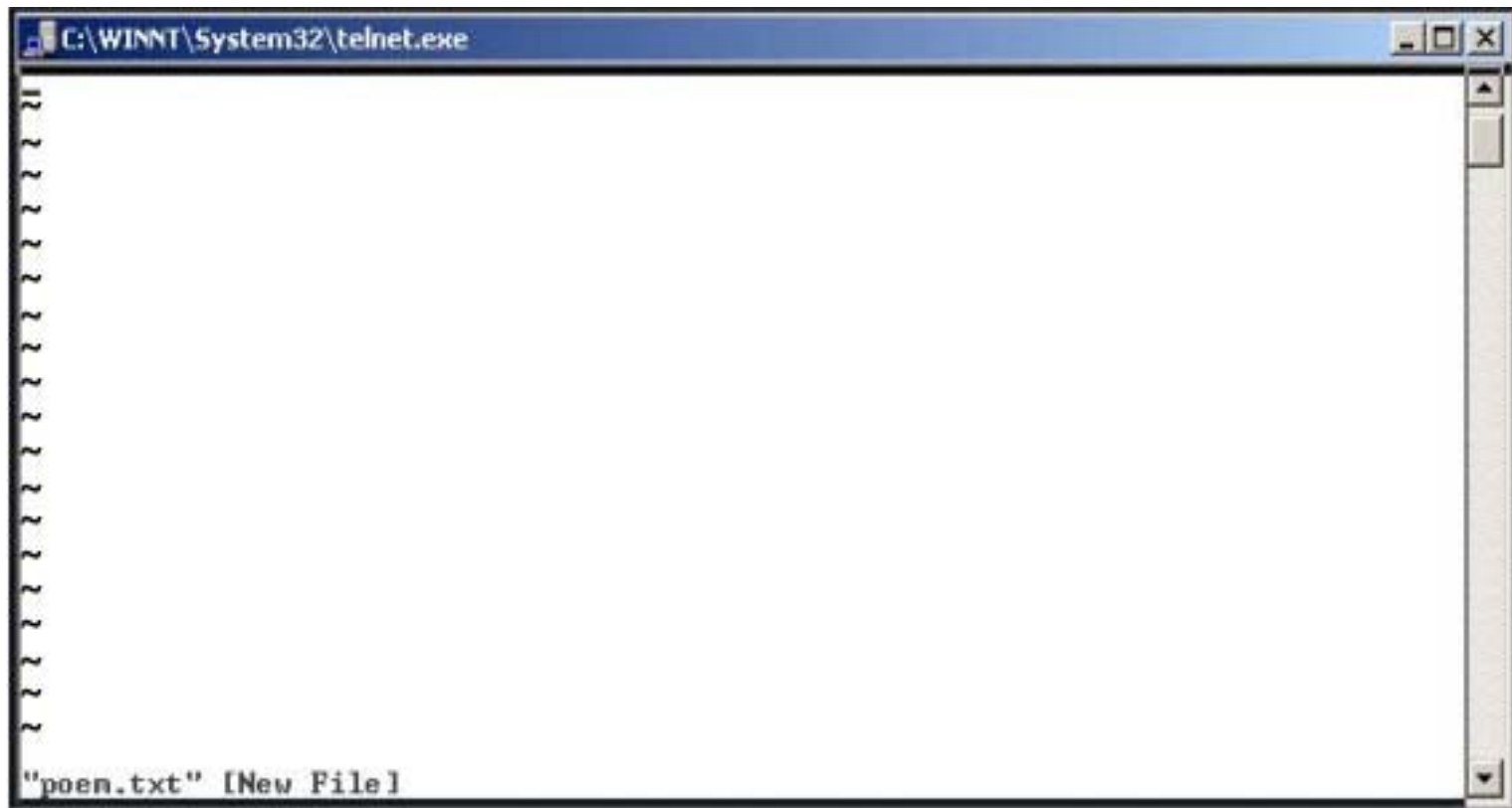
---



# Fundamentos de vi

---

Para abrir un editor **vi** se usa el comando:  
`/home/mickeymouse]$ vi poem.txt`





# Inserción de Texto en el Editor vi

---

- Se crean los documentos al ingresar texto en el editor **vi**
- No se puede ingresar texto directamente en la ventana del editor **vi**, dado que el texto no aparecería en la pantalla
- Para iniciar el ingreso de texto en este editor, se tiene que ejecutar un comando, conocido como *comando de inserción*.
- Texto Ejemplo:

**mickeymouse**

# Comandos de Inserción vi

| Comando  | Propósito                         | Ejemplo  |
|----------|-----------------------------------|--|
| <b>i</b> | Inserta a la izquierda del cursor | Si el cursor está en <code>'y'</code> , inserta el texto a la izquierda de <code>'y'</code> .<br><b>Resultado:</b> micke <b>key</b> mouse  |
| <b>a</b> | Inserta a la derecha del cursor   | Si el cursor está en <code>'y'</code> , entonces inserta el texto a la derecha de <code>'y'</code> .<br><b>Resultado:</b> mickey <b>key</b> mouse  |
| <b>I</b> | Inserta al inicio de la línea     | Inserta el texto a la izquierda de la primera letra <code>'m'</code> de la palabra <code>'mickeymouse'</code> , sin importar la posición del cursor.<br><b>Resultado:</b> <b>key</b> mickeymouse |

# Comandos de Inserción vi... 2

| Comando  | Propósito   | Ejemplo   |
|----------|---|---|
| <b>A</b> | Inserta al final de la línea                              | Inserta el texto a la derecha de la primera letra 'm' de la palabra 'mickeymouse', sin importar la posición del cursor.<br><b>Resultado:</b> mickeymousekey |
| <b>o</b> | Abre una nueva línea y agrega texto bajo la línea actual  | Ayuda a escribir una nueva línea bajo 'mickeymouse'.<br><b>Resultado:</b> mickeymouse<br><línea en blanco abierta para insertar texto>                      |
| <b>O</b> | Abre una nueva línea y agrega texto sobre la línea actual | Ayuda a escribir una nueva línea sobre 'mickeymouse'.<br><b>Resultado:</b> <línea en blanco abierta para insertar texto ><br>mickeymouse                    |

# Modos de Inserción en el Editor vi



A screenshot of a Windows telnet window titled "C:\WINNT\System32\telnet.exe". The window displays the text of a poem in a monospaced font:

```
I wanna brek free  
From the sackles of this society,  
From the rules and regulations governing it,  
From its prevailing age-old.
```

Below the poem, there is a vertical column of yellow characters, which are the escape sequence for entering insert mode in vi. At the bottom left of the window, the text "-- INSERT --" is displayed, indicating that the editor is currently in insert mode. The window has standard Windows controls (minimize, maximize, close) in the top right corner and a vertical scrollbar on the right side.

# Modos de Inserción vi en el Editor ... 2

---

- Considere que se quiere agregar `'prototype'` al final de la línea actual
- Esto significa que se quiere agregar texto a la línea actual
- Al terminar el ingreso inicial de las cuatro líneas del poema, el cursor estará posicionado al final de la última línea
- Ingrese el comando **A**
- Note que el cursor se ha movido a una posición luego del final

# Modos de Inserción en el Editor vi... 3

---

I wanna brek free

From the sackles of this society,

From the rules and regulations governing it,

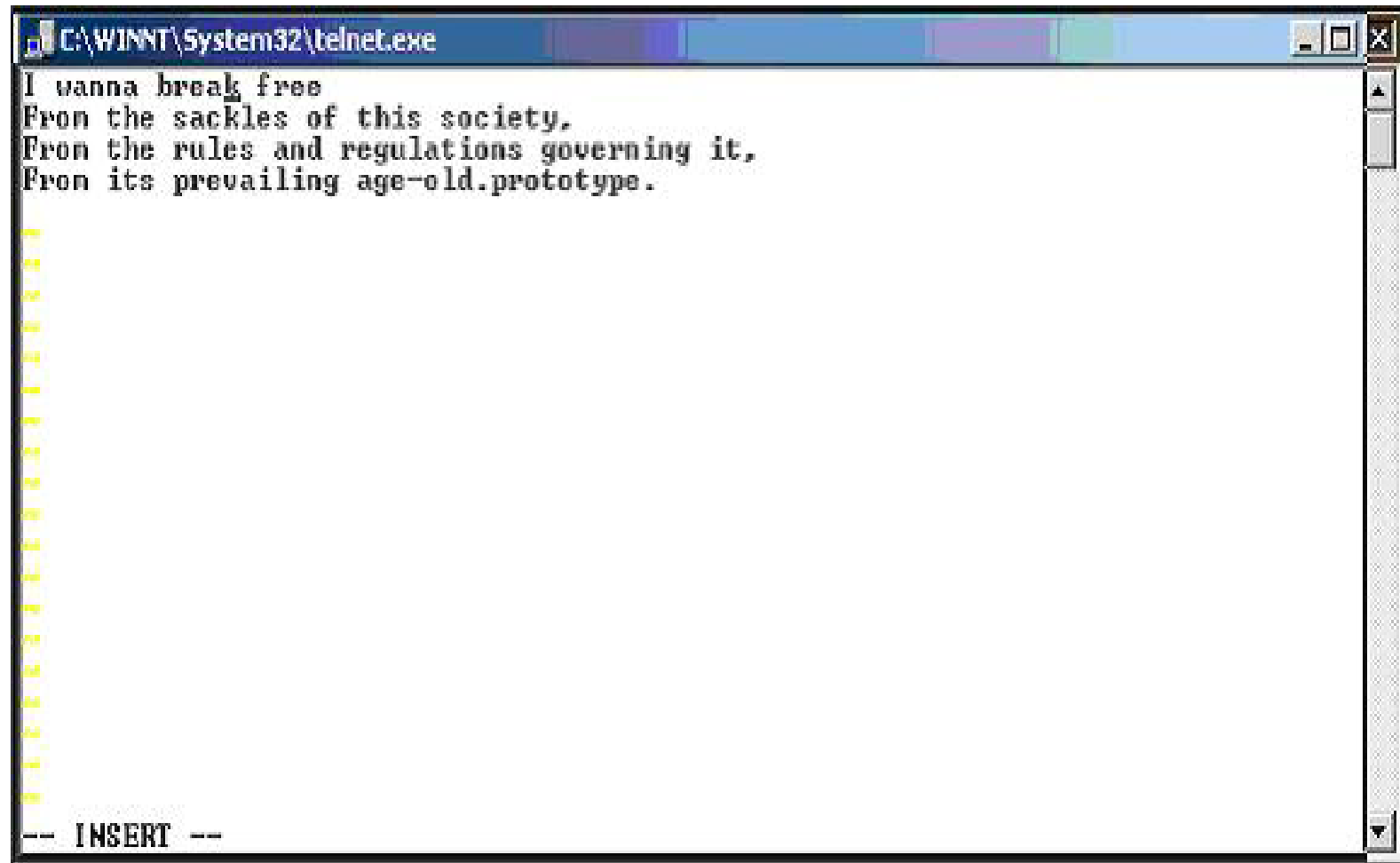
From its prevailing age-old. prototype

- Para cambiar “**brek**” por “**break**” se puede usar cualquiera de los comandos de inserción: a, I, A ó l
- Usando las teclas del cursor para moverse por la pantalla, se posicionará el cursor bajo la letra 'e' de '**brek**' y se ingresará a ,el comando para insertar.
- El editor vi ahora está en modo de inserción. Se ingresa la letra 'a'



# Luego de Agregar “prototype”

---



A screenshot of a Windows telnet session window. The title bar reads "C:\WINNT\System32\telnet.exe". The window contains a poem in a monospaced font:

```
I wanna break free  
From the shackles of this society,  
From the rules and regulations governing it,  
From its prevailing age-old.prototype.
```

Below the poem, there is a vertical column of yellow dashes. At the bottom left of the window, the text "-- INSERT --" is displayed.

# Guardar un Archivo en el Editor vi

---

- Se puede agregar texto y hacer correcciones al texto usando los comandos de inserción.
- Se usan los comandos para guardar disponibles en **vi**, a través de **ex**, para guardar en el almacenamiento secundario
- Los comandos para guardar mostrados en la tabla son comandos para guardar de **ex**, que se usan en **vi**



# Comandos para Guardar

---

| Comando   | Propósito                                   |
|-----------|---|
| <b>w</b>  | Guardar y continuar el trabajo              |
| <b>wq</b> | <b>Guardar y salir de</b> <small>vi</small> |
| <b>q!</b> | Salir sin guardar el trabajo                |

# Eliminación de Texto en vi

---

- Para eliminar texto se mueve el cursor hasta el carácter punto y se presiona '**x**' para eliminar el carácter
- Asuma que ahora se quiere eliminar la palabra '**sackles**' en la segunda línea del mismo ejemplo
- Se mueve el cursor hasta la primera letra de la palabra, la letra **s** en este caso, y luego se ingresa el comando **dw**. La palabra será eliminada

# Comandos de Eliminación de Texto en vi

| Comando   | Propósito  | Ejemplo  |
|-----------|--|--|
| <b>x</b>  | Elimina un carácter, donde el cursor está ubicado.                       | Si el cursor esta en 'k', entonces elimina 'k'.<br><b>Resultado:</b> Hi miceymouse |
| <b>dw</b> | Elimina desde la posición actual del cursor hasta el final de la palabra | Si el cursor esta en 'y', entonces elimina 'ymouse'.<br><b>Resultado:</b> Hi micke |
| <b>dd</b> | Elimina la línea actual  | Si el cursor está en 'Hi mickeymouse', entonces elimina 'Hi mickeymouse'.          |
| <b>D</b>  | Elimina desde la posición actual del cursor hasta el final de la línea   | Si el cursor está en 'i' de 'Hi' elimina 'i mickeymouse'<br><b>Resultado:</b> H    |

# Eliminar Texto – Comandos x y dw

---



A screenshot of a Windows telnet window. The title bar reads "C:\WINNT\System32\telnet.exe". The window contains the following text:

```
I wanna break free  
From the of this society,  
From the rules and regulations governing it,  
From its prevailing age-old prototype.
```

Below the text, there is a vertical column of yellow dashes, likely representing a list or a series of commands being entered or displayed.

# Modificadores de Comandos

---

- **vi** proporciona una característica llamada modificadores de comando para eliminar más de una letra, palabra o línea usando el mismo comando a fin de ahorrar tiempo
- Los modificadores de comando mejoran y aumentan el poder de los comandos de eliminación de texto
- Texto Ejemplo:

**Hello Mickeymouse**

**Hello Donald**

**Hello Asterik**

# Modificadores de Comandos... 2

| Comando    | Propósito  | Ejemplo  |
|------------|--|--|
| <b>nx</b>  | Elimina <b>n</b> caracteres a partir de la posición del cursor | Si el cursor está en 'c' de 'Hi <b>Mickeymouse</b> ' en la primera línea, entonces <b>3x</b> elimina 'cke'.<br><b>Resultado:</b> Hello Miymouse<br>Hello Donald<br>Hello Asterik |
| <b>dnw</b> | Elimina <b>n</b> palabras de la posición actual del cursor     | Si el cursor está en 'e' de 'Hello' de la primera línea, entonces <b>d2w</b> elimina 'ello' y 'Mickeymouse'.<br><b>Resultado:</b> H<br>Hello Donald<br>Hello Asterik             |
| <b>ndd</b> | Elimina <b>n</b> líneas de la posición actual del cursor       | Si el cursor está en 'H' de 'Hello' en la segunda línea, entonces <b>2dd</b> elimina 'Hello Donald' y 'Hello Asterik'.<br><b>Resultado:</b> Hello Mickeymouse                    |

# Modificadores de Comandos... 3

| Comando | Propósito  | Ejemplo  |
|---------|--|--|
| DG      | Elimina desde la posición actual del cursor hasta el final de dicha línea y lleva al cursor hasta el final del archivo | Si el cursor está en 'M' de 'Mickeymouse' en la primera línea, entonces elimina 'Mickeymouse', y lleva el cursor al final del archivo.<br><b>Resultado:</b> Hello<br>Hello Donald<br>Hello Asterik                 |
| DnG     | Elimina la línea donde el cursor está presente, y mueve el cursor al inicio de la <sup>n</sup> ésima línea             | Si la posición actual del cursor es 'H' de 'Hello Donald', entonces D1G elimina 'Hello Donald' y lleva el cursor a la letra 'H' de 'Hello Mickeymouse'.<br><b>Resultado:</b><br>Hello Mickeymouse<br>Hello Asterik |

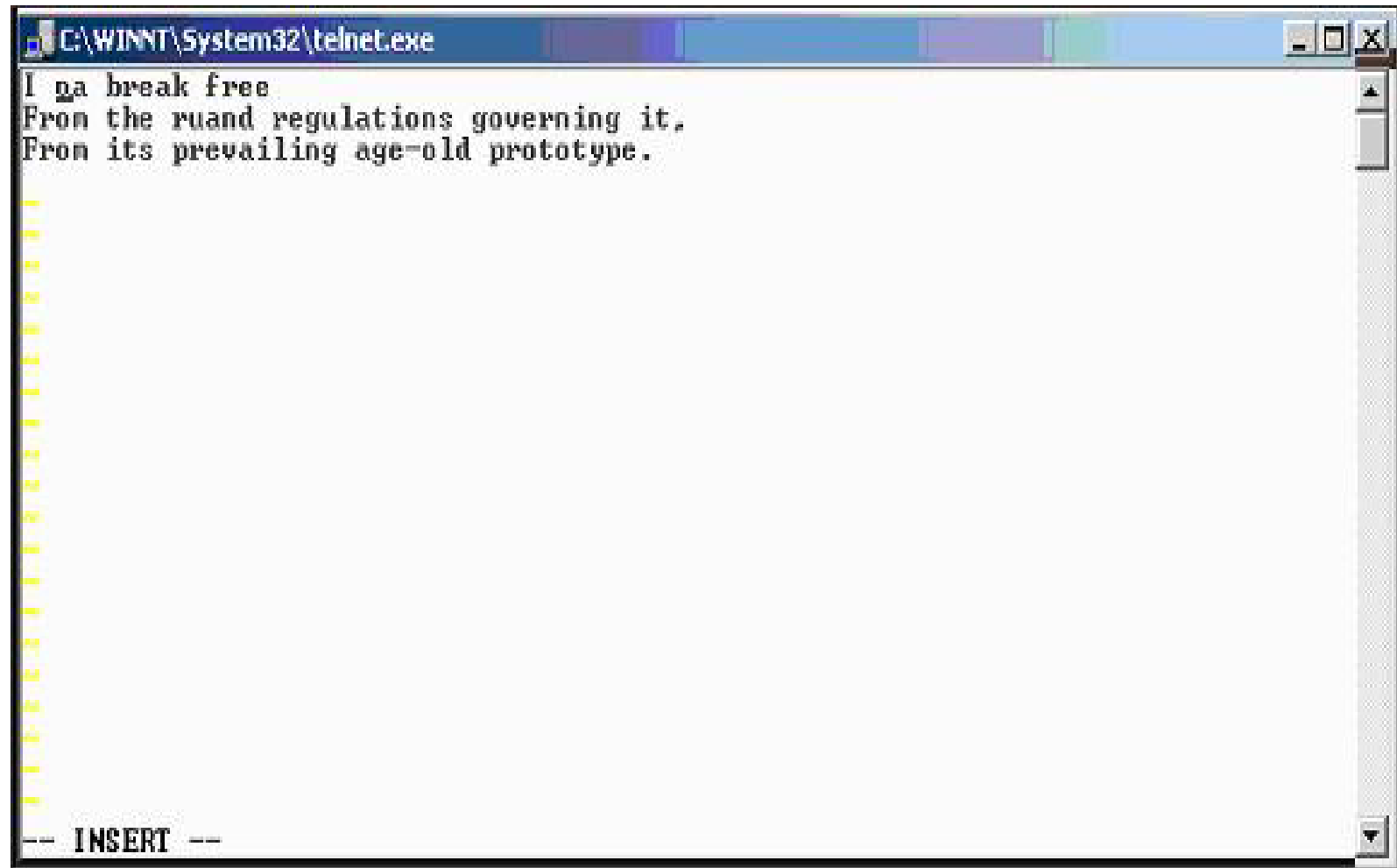
# Modificadores de Comandos... 4

| Comando | Propósito  | Ejemplo   |
|---------|--|---|
| D\$     | Elimina a partir de la posición actual hasta el final de la línea                | Si el cursor está en 's' de 'Asterik', entonces elimina 'sterik'.<br><b>Resultado:</b> Hello Mickeymouse<br>Hello Donald<br>Hello A                 |
| dn\$    | Elimina desde la posición actual del cursor hasta el final de la línea<br>n dada | Si el cursor está en 'y' de 'Hello Mickeymouse', entonces d2\$ elimina 'ymouse' y 'Hello Donald'.<br><b>Resultado:</b> Hello Micke<br>Hello Asterik |



# Modificador de Comando (3x)

---



A screenshot of a Windows telnet window titled "C:\WINNT\System32\telnet.exe". The window displays a poem in a monospaced font. The poem is: "I ga break free  
From the ruand regulations governing it,  
From its prevailing age-old prototype." The text is left-aligned. The window has a standard Windows XP-style title bar with minimize, maximize, and close buttons. On the right side, there is a vertical scrollbar. At the bottom left of the window, the text "-- INSERT --" is visible. The background of the window is white, and the text is black.

```
C:\WINNT\System32\telnet.exe
I ga break free
From the ruand regulations governing it,
From its prevailing age-old prototype.

-- INSERT --
```

# Mas Modificadores de Comandos

---

- **Para eliminar la palabra 'ruand regulations'**

Se coloca el cursor al inicio de la palabra, la letra ``r'` en este caso, y luego se ingresa el comando `2dw`. Las dos palabras se eliminan.

- **Para eliminar más de una línea a la vez**

Para eliminar las dos primeras líneas del poema, se coloca el cursor bajo `'I'` en la primera línea y se ingresa el comando `d2$`. Las líneas serán eliminadas

# Moverse en la pantalla - Comandos

| Comando  | Propósito  | Ejemplo   |
|----------|--|---|
| <b>h</b> | Un espacio a la izquierda de la posición actual del cursor | Si el cursor está en <code>'c'</code> de <code>"Hockey"</code> , lleva al cursor a la <code>'o'</code> de <code>"Hockey"</code> .             |
| <b>l</b> | Un espacio a la derecha de la posición actual del cursor   | Si el cursor está en <code>'F'</code> de <code>"Football"</code> , lleva el cursor a la primera <code>'o'</code> de <code>"Football"</code> . |
| <b>j</b> | Un espacio debajo de la posición actual del cursor         | Si el cursor está en <code>'r'</code> de <code>'Cricket'</code> , lleva el cursor a la primera <code>'o'</code> de <code>'Football'</code> .  |
| <b>k</b> | Un espacio arriba de la posición actual del cursor         | Si el cursor está en <code>'k'</code> de <code>'Hockey'</code> lleva el cursor a <code>'t'</code> de <code>'Football'</code> .                |

# Movimiento del Curso – Por palabra

| Comando       | Propósito  | Ejemplo   |
|---------------|--|---|
| <b>e or E</b> | Mueve el cursor al final de la siguiente palabra cuando el cursor está al final de la palabra actual. Caso contrario al final de la misma palabra        | Si el cursor está en 'c' de "welcome", lleva el cursor a 'e' al final de "welcome". Si el cursor está en 'o' de "to", lleva el cursor a 'x' de "Linux". |
| <b>w</b>      | Mueve el cursor al inicio de la siguiente palabra  | Si el cursor está en 'o' de "welcome", lleva el cursor a 't' de "to".   |
| <b>B</b>      | Mueve el cursor al inicio de la palabra anterior sólo cuando el cursor está al inicio de la palabra actual, caso contrario al inicio de la misma palabra | Si el cursor está en 'm' de "welcome", lleva el cursor a 'w' de "welcome". Si el cursor está en 't' de 'to', lleva el cursor a 'w' de "welcome".        |

# Movimiento del Curso – Por línea

| Comando           | Propósito   | Ejemplo   |
|-------------------|---|---|
| <b>0</b>          | Mueve al inicio de la línea   | Si el cursor está en 'l' de 'line' en la primera línea, lleva el cursor a 'T' de 'This'.                    |
| <b>^</b>          | Mueve a la primera palabra de la línea  | Si el cursor está en 'e' de 'test' en la primera línea, lleva el cursor a 'T' de 'This'                     |
| <b>\$</b>         | Mueve al final de la línea  | Si el cursor está en 'i' de 'This' en la primera línea, lleva el cursor a 'e' de 'line'                     |
| <b>&lt;CR&gt;</b> | Mueve al inicio de la siguiente línea. (CR viene de 'carriage return' - retorno de carro, equivale a la tecla <Enter> ) | Si el cursor está en 'n' de 'line' en la primera línea, lleva el cursor a 'T' de 'This' en la segunda línea |

# Movimiento del Curso – Archivo

---

| Comando   | Propósito                                    | Ejemplo  |
|-----------|--|--|
| <b>G</b>  | Mueve al primer carácter de la última línea  | Si el cursor está en <code>'i'</code> de <code>'is'</code> en la primera línea, lleva el cursor a <code>'T'</code> de <code>'This'</code> en la segunda línea.   |
| <b>1G</b> | Mueve al primer carácter de la primera línea | Si el cursor está en <code>'s'</code> de <code>'test'</code> en la segunda línea, lleva el cursor a <code>'T'</code> de <code>'This'</code> en la primera línea. |

# Movimiento del Curso – Pantalla

---

| Comando                     | Propósito                              |
|-----------------------------|--|
| <code>&lt;ctrl&gt; f</code> | Avanza (una pantalla completa)         |
| <code>&lt;ctrl&gt; b</code> | Retrocede (una pantalla completa)      |
| <code>&lt;ctrl&gt; d</code> | Desplaza hacia abajo (media pantalla)  |
| <code>&lt;ctrl&gt; u</code> | Desplaza hacia arriba (media pantalla) |

# Movimiento del Curso – Otros Comandos

| Comando               | Propósito                           | Ejemplo   |
|-----------------------|-------------------------------------|---|
| <b>NG</b>             | Al número de línea <b>n</b>         | Por ejemplo, <b>3G</b> lleva el cursor a la tercera línea, ' <b>This is second line</b> '.                  |
| <b>&lt;ctrl&gt; G</b> | El número de línea actual           | Si el cursor está en ' <b>This is second line</b> ', se muestra número de línea <b>3 of 5--60%--col 1</b> . |
| <b>%</b>              | La llave o corchete correspondiente | Lleva el cursor de la llave ' <b>{</b> ' de apertura a su correspondiente llave ' <b>}</b> ' de cierre.     |



# Movimiento del Curso – Otros Comandos... 2

| Comando                                    | Propósito   | Ejemplo  |
|--|---|--|
| <b>n  </b><br>(n seguido por el símbolo  ) | Mueve el cursor a la columna n, donde n es un entero. | Si el cursor está en 's' de 'second' en la tercera línea, entonces 3   lleva al usuario a 'i' de 'This' en la tercera línea  |
| <b>n1</b><br>(n seguido por 1)             | Mueve el cursor n columnas a la derecha del cursor    | Si el cursor está en 's' de 'second' en la tercera línea, entonces 41 lleva al usuario a 'n' de "second" en la tercera línea |

# Modificación de Texto – Deshacer cambios

| Comando    | Propósito   | Ejemplo   |
|------------|---|---|
| <b>u</b>   | Deshace el último comando   | Si se agrega ' <b>key</b> ' a ' <b>Atomica</b> ', revierte al texto original ' <b>Atomica</b> '.  |
| <b>U</b>   | Deshace los cambios en la línea actual  | Si se cambia la línea de ' <b>Atomica</b> ' a ' <b>Cell</b> ', revierte al texto previo ' <b>Atomica</b> '  |
| <b>:e!</b> | Editar de nuevo. Restaura el texto al estado que tenía la última vez que se grabó | Si se agrega ' <b>1</b> ' a ' <b>Atomica</b> ' haciendolo ' <b>Atomical</b> ' y se graba, este comando revertirá al texto anterior ' <b>Atomica</b> ' |

# Otros comandos de modificación de texto

| Comando  | Propósito   | Ejemplo  |
|----------|---|--|
| <b>r</b> | Reemplaza el carácter donde está situado el cursor, con una letra | Si el cursor está en 'S' de 'Server', entonces ingresar <b>rC</b> reemplaza 'S' con 'C' para hacer 'Cerver'.<br><b>Resultado:</b> This Cerver<br>This Client     |
| <b>R</b> | Reemplaza el texto con el nuevo texto                             | Si el cursor está en 'v' de 'Server', ingresar <b>key R</b> cambia el texto 'key' por 'ver' para hacer 'Serkey'.<br><b>Resultado:</b> This Serkey<br>This Client |

# Otros Comandos de Modificación de Texto... 2

| Comando    | Propósito  | Ejemplo   |
|------------|--|---|
| <b>cw</b>  | Cambia el texto de la palabra actual                             | Si el cursor está en ' <b>Server</b> ', ingresar <b>cw key</b> para hacer ' <b>This Key</b> '.<br><br><b>Resultado:</b> This key<br>This Client   |
| <b>c\$</b> | Cambia el texto de la posición actual hasta el final de la línea | Ayuda a eliminar letras desde ' <b>r</b> ' en ' <b>Server</b> ' hasta el final de la línea y cambiarlas por el texto ' <b>key</b> '. Esto hace la nueva palabra ' <b>This Sekey</b> '.<br><br><b>Resultado:</b> This Sekey<br>This Client |

# Otros Comandos de Modificación de Texto... 3

| Comando     | Propósito  | Ejemplo  |
|-------------|--|--|
| <b>cnw</b>  | Cambia las siguientes <b>n</b> palabras.<br>(Igual como <b>ncw</b> ) | Por ejemplo, <b>c2w</b> ayuda a cambiar el texto ' <b>This Server</b> ' por otro texto diferente, tal como ' <b>That Cell</b> '.<br><br><b>Resultado:</b> That Cell<br>This Client   |
| <b>cn\$</b> | Cambia hasta el final de la línea <b>n</b>                           | Si el cursor está en ' <b>r</b> ' de ' <b>Server</b> ', entonces <b>c2\$</b> ayuda a cambiar el texto ' <b>rver</b> ' por otro texto diferente, como ' <b>key</b> ' y además la siguiente línea ' <b>This Client</b> ' con un texto, tal como ' <b>This Router</b> '.<br><br><b>Resultado:</b> This Sekey<br>This Router |

# Otros Comandos de Modificación de Texto... 4

| Comando | Propósito                         | Ejemplo  |
|---------|-----------------------------------|--|
| c       | Cambia hasta el final de la línea | <p>Si el cursor está en 'e' de 'Client' y el nuevo texto es 'key', entonces se vuelve 'Clikey'.</p> <p><b>Resultado:</b> This Server<br/>This Clikey</p> |
| cc      | Cambia la línea actual            | <p>Cambia toda la línea, por ejemplo, de 'This Server' a 'That Server'.</p> <p><b>Resultado:</b> That Server<br/>This Client</p>                         |

# Otros comandos de modificación de texto... 5

| Comando   | Propósito   | Ejemplo   |
|-----------|---|---|
| <b>s</b>  | Sustituye el carácter actual por el texto ingresado         | Cambia la letra 'C' de 'Client' con 'P' para hacer 'Plient'.<br><b>Resultado:</b> This Server<br>This Plient                                      |
| <b>ns</b> | Sustituye el texto ingresado en los siguientes n caracteres | Ingresando 3s en el texto 'Server' con el cursor en 'e' y el nuevo texto 'key', lo hace 'Skeyer'.<br><b>Resultado:</b> This Skeyer<br>This Client |

# Otros comandos de modificación de texto... 6

---

| Comando  | Propósito                   | Ejemplo  |
|----------|-----------------------------|--|
| <b>S</b> | Reemplaza la línea completa | Si el cursor está bajo 'C' de 'Client' y si se ingresa <b>S</b> seguido por <b>Server</b> , cambiará la línea a ' <b>Server</b> '. Al presionar <b>S</b> , se elimina toda la línea. |



# Copiar y Pegar Texto

---

- **Copiar texto involucra los siguientes tres pasos:**

- Copiar el texto a un buffer.

Un buffer es una ubicación de almacenamiento temporal para guardar texto

- Mover el cursor al lugar de destino
- Pegar (colocar) el texto del buffer

- **Texto Ejemplo:**

**I am Happy**

**I am Wise**

# Comandos de Copia

| Comando    | Propósito   | Ejemplo   |
|------------|---|---|
| <b>yy</b>  | Mueve una copia de la línea actual al buffer sin nombre | Si el cursor esta en 'I' de 'I am Happy', entonces copia 'I am Happy' a un buffer sin nombre            |
| <b>y</b>   | Mueve una copia de la línea actual al buffer sin nombre | Si el cursor esta en 'I' de 'I am Happy', entonces copia 'I am Happy' a un buffer sin nombre.           |
| <b>nyy</b> | Mueve las $n$ líneas siguientes al buffer sin nombre    | Si el cursor esta en 'I am Happy', entonces 2yy copia 'I am Happy' y 'I am Wise' a un buffer sin nombre |
| <b>nY</b>  | Mueve las $n$ líneas siguientes al buffer sin nombre    | Si el cursor esta en 'I am Happy', entonces 2Y copia 'I am Happy' y 'I am Wise' a un buffer sin nombre  |

# Comandos de Copia

| Comando    | Propósito   | Ejemplo   |
|------------|---|---|
| <b>yw</b>  | Mueve un palabra al buffer sin nombre             | Si el cursor está en 'H' de 'Happy', entonces copia 'Happy' a un buffer sin nombre.                                   |
| <b>ynw</b> | Mueve n palabras al buffer sin nombre             | Si el cursor está en 'a' de 'am' en la primera línea, entonces <b>y2w</b> copia 'am' y 'Happy' a un buffer sin nombre |
| <b>nyw</b> | Mueve n palabras al buffer sin nombre             | Si el cursor está en 'a' de 'am' en la primera línea, entonces <b>2yw</b> copia 'am' y 'Happy' a un buffer sin nombre |
| <b>y\$</b> | Mueve la posición del cursor al final de la línea | Si el cursor está en 'I' de 'I am Happy' se mueve a 'y', que está al final de esa línea.                              |

# Comandos de Pegar

---

| Comando   | Propósito   | Ejemplo  |
|-----------|---|--|
| <b>P</b>  | Pega del buffer sin nombre a la derecha del cursor            | Pega, por ejemplo, 'Happy' almacenado en el buffer sin nombre, al lado derecho del cursor          |
| <b>P</b>  | Pega del buffer sin nombre a la izquierda del cursor          | Pega, por ejemplo, 'Happy' almacenado en el buffer sin nombre, al lado izquierdo del cursor        |
| <b>nP</b> | Pega n copias del buffer sin nombre a la izquierda del cursor | <b>2P</b> pega 'Happy' almacenado en el buffer sin nombre, dos veces al lado izquierdo del cursor. |

# Comandos de Copia de Buffer con Nombre

---

| Comando   | Propósito   | Ejemplo   |
|-----------|---|---|
| "<char>yy | Mueve la línea actual al buffer con nombre <char>   | Si el cursor está en 'T' de 'This' en la primera línea, copia 'This is Cat' al buffer con nombre, por ejemplo, m  |
| "<char>Y  | Mueve la línea actual al buffer con nombre <char>   | Si el cursor está en 'T' de 'This' en la primera línea, copia 'This is Cat' al buffer con nombre, por ejemplo, m. |
| "<char>yw | Mueve la palabra actual al buffer con nombre <char> | Si el cursor está en 'C' de 'Cat' en la primera línea, copia 'Cat' al buffer con nombre, por ejemplo, m.          |

# Comandos de Copia de Buffer con Nombre..2

| Comando                              | Propósito  | Ejemplo  |
|--------------------------------------|--|--|
| "<char>yw<br>(<char>en<br>mayúscula) | Agrega la<br>palabra al<br>contenido del<br>buffer con<br>nombre<br><char> | Si el cursor está en 'D' de 'Dog'<br>en la primera línea, agrega 'Dog'<br>al buffer con nombre, por<br>ejemplo, <b>M</b> , que ya contenía<br>'Cat'. |
| "<char>y2w                           | Mueve las dos<br>siguientes<br>palabras al<br>buffer <char>                | Si el cursor está en 'T' de<br>'This' en la primera línea,<br>entonces <b>my2w</b> copia 'This<br>is' al buffer con nombre, <b>m</b> .               |
| "<char>p                             | Pega desde el<br>buffer con<br>nombre <char><br>a la derecha<br>del cursor | Pega el buffer con nombre, por<br>ejemplo, <b>m</b> , que contiene por<br>ejemplo, 'Cat' al lado derecho<br>del cursor.                              |

# Comandos de Copia de Buffer con Nombre..3

| Comando               | Propósito  | Ejemplo   |
|-----------------------|--|---|
| <b>&lt;char&gt;p</b>  | Pega desde el buffer con nombre <b>&lt;char&gt;</b> a la derecha del cursor            | Pega el valor del buffer con nombre, que contiene, por ejemplo 'Cat' al lado derecho del cursor.                            |
| <b>&lt;char&gt;nP</b> | Pega n copias desde el buffer con nombre <b>&lt;char&gt;</b> a la izquierda del cursor | <b>M3P</b> pega tres copias del buffer con nombre <b>M</b> , que contiene, por ejemplo, 'Dog' al lado izquierdo del cursor. |

# Cortar y Pegar Texto

---

- Cortar y pegar texto consiste de los siguientes tres pasos:
  - Eliminar el texto hacia un buffer con o sin nombre
  - Mover el cursor a la ubicación destino
  - Pegar el buffer con nombre o sin nombre
- El proceso es el mismo cuando se copia, pero con un cambio en el primer paso, de copiar a eliminar



# Comandos para Cortar y Pegar

| Comando     | Propósito  | Ejemplo   |
|-------------|--|---|
| <b>bdd</b>  | Elimina la línea y la coloca en el buffer con nombre <b>b</b>    | Si el cursor está en ' <b>M</b> ' de ' <b>My Book</b> ', elimina dicha línea y la copia al buffer con nombre <b>b</b> .           |
| <b>B2dd</b> | Elimina dos líneas y las coloca en el buffer con nombre <b>B</b> | Si el cursor está en ' <b>M</b> ' de ' <b>My Book</b> ', elimina 'My Book' y 'My Pen' y copia ambos al buffer con nombre <b>B</b> |
| <b>dw</b>   | Elimina una palabra y la coloca en el buffer sin nombre          | Si el cursor está en ' <b>P</b> ' de ' <b>My Pen</b> ', elimina la palabra ' <b>Pen</b> ' y la copia al buffer sin nombre         |

# Comandos de Búsqueda

| Comando   | Propósito   | Ejemplo   |
|-----------|---|---|
| <b>fc</b> | Encuentra el siguiente carácter <b>c</b> a la derecha del cursor en la misma línea                    | Si el cursor está en <b>'C'</b> de <b>'Client'</b> y se usa <b>fa</b> , el cursor se mueve hasta <b>'a'</b> de <b>'Machine'</b> de la primera línea                     |
| <b>Fc</b> | Encuentra el siguiente carácter <b>c</b> a la izquierda del cursor en la línea                        | Si el cursor está bajo <b>'c'</b> de <b>'Machine'</b> en la segunda línea y se usa <b>FT</b> , el cursor se mueve hasta <b>'T'</b> de <b>"The"</b> de la segunda línea. |
| <b>tc</b> | Encuentra el carácter antes del siguiente carácter <b>c</b> a la derecha del cursor en la misma línea | Si el cursor está en <b>'C'</b> y se usa <b>ta</b> , el cursor se mueve hasta <b>'M'</b> de <b>'Machine'</b> de la primera línea  |

# Comandos de Búsqueda... 2

| Comando   | Propósito   | Ejemplo  |
|-----------|---|--|
| <b>Tc</b> | Encuentra el carácter después del siguiente carácter <b>c</b> a la derecha del cursor en la misma línea | Si el cursor está bajo 'c' de <b>'Machine'</b> en la segunda línea y se usa <b>FS</b> , el cursor se mueve hasta 'e' luego de 'S' de <b>"Server"</b> de la segunda línea.                          |
| <b>;</b>  | Repite el último f, F, t y T hacia adelante   | Si el cursor está en 'C', y se usa <b>fe</b> , el cursor se mueve al primer 'e' a la derecha. Al presionar <b>;</b> va al siguiente <b>e</b> , si lo encuentra, en la misma línea hacia la derecha |
| <b>,</b>  | Repite el último f, F, t y T hacia atrás  | Igual que <b>;</b> pero invierte la dirección del comando original   |

# Comandos de Búsqueda de Cadenas

| Comando        | Propósito  | Ejemplo  |
|----------------|--|--|
| <b>/cadena</b> | Encuentra la siguiente ocurrencia de <b>cadena</b> | Si el cursor está en 'C' de la primera palabra 'Client' en la primera línea, ingresando <b>/Client</b> se moverá el cursor hasta 'C' de 'Client', que es la última palabra en la línea actual (la siguiente ocurrencia) en el archivo. |
| <b>?cadena</b> | Encuentra la última ocurrencia de <b>cadena</b>    | Si el cursor está en 'C' de la primera palabra 'Client' en la primera línea, ingresando <b>?Client</b> se moverá el cursor a 'C' de 'Client', la segunda palabra en la segunda línea (la última ocurrencia) en el archivo.             |
| <b>n</b>       | Repite el último comando / o ? hacia adelante      | Para la <b>/cadena</b> usada antes, <b>n</b> repite y encuentra el siguiente <b>Client</b> , que es la segunda palabra en la segunda línea.  |
| <b>N</b>       | Repite el último comando / o ? hacia atrás         | Igual que <b>n</b> , pero invierte la dirección del comando original.  |

# Resumen

---

- Se explicó cómo el editor visual vi trabaja en Linux
- Se establecieron las diferentes categorías de comandos
- Se explicó el uso de los comandos para editar archivos

## Laboratorio del Editor vi