**Taller #1**
Instituto Tecnológico de Costa Rica
Introducción a los Sistemas Embebidos - CE 5303
Prof. Luis Alonso Barboza Artavia

**Estudiante:** Esteban Josué Alvarado Vargas
**Carné:** 2018109336

---

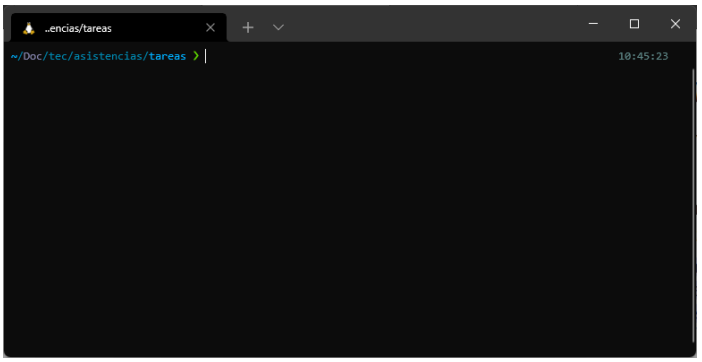
# Uso de Linux

### Pregunta 1.1
¿Qué es una distribución Linux y qué puede incluir?

Una **distribución** (también conocida como *distro*) es un conjunto de utilidades, herramientas y programas basados en el núcleo de *GNU/Linux* [1]. Las distribuciones pueden incluir aplicaciones, utilidades y controladores, además, algunas ofrecen procedimientos de instalación y actualización [2]. Existen distribuciones para toda clase de computadores de escritorio, laptops, dispositivos de red, servidores, dispositivos móviles, sistemas embebidos y muchos otros.

### Pregunta 1.2
¿Qué es un *Shell*?

El ***Shell*** es un intérprete de comandos que provee una interfaz interactiva entre el usuario y el sistema operativo. El usuario ingresa comandos en la línea de comandos; estos son interpretados por el *Shell* y son enviados como instrucciones al sistema operativo. Además, los comandos pueden escribirse en un fichero script para ser ejecutados consecutivamente, como un programa. El Shell tiene la capacidad de manipular las entradas y salidas, incluso puede ejecutar procesos en segundo plano [3]. Algunos de los Shell para Linux son: `BASH`, `ZSH`, `FISH` y `TCSH`.




### Pregunta 1.3
¿Cuál es la diferencia entre una ruta relativa y una ruta absoluta a un archivo?

Las rutas indican la ubicación de un archivo o directorio dentro del sistema de archivos. Las rutas pueden ser absolutas o relativas. Una **ruta absoluta** es la ruta completa de un fichero comenzando por el directorio `root` [3].

Ejemplo: `/home/ealvarado/Documents/tec/asistencias/tareas`

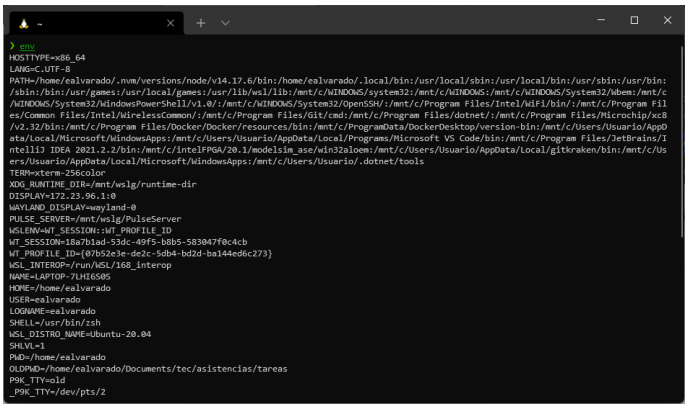
Una **ruta relativa** inicia desde el directorio de trabajo actual (`wd`); es la ruta de un archivo relativa al directorio de trabajo [3].

Ejemplo: `/tec/asistencias/tareas` **`* wd: Documents/`**

### Pregunta 1.4

¿Qué información brinda el comando `env`?

El comando **`env`** permite imprimir en el Shell todas las variables de entorno que están definidas. Las variables de entorno son aquellas que están disponibles para ser referenciadas por cualquier programa, su valor es copiado a cualquier otro programa o Shell creado desde su entorno [4].




### Pregunta 1.5

¿Para qué son útiles los pipes de UNIX? Muestre un ejemplo de su utilidad.

Los **pipes** (tuberías), son un mecanismo de los sistemas operativos que permite *redirigir* la salida estándar de un proceso hacia la entrada de otro proceso [1]. Este proceso de llama *piping*, porque genera un *pipeline* por donde se recibe una entrada por un extremo y los transmite, ya modificados, por el otro. Por un extremo del pipe se escribe y por el otro se lee. Los pipes son creados y destruidos por el *kernel* [2].

Este mecanismo permite *combinar comandos para ejecutar procesos complejos* desde la línea de comandos. Resulta ser bastante eficiente, ya que, la salida de un programa se pasa directamente como entrada al segundo programa, sin tener que escribir en los archivos temporales [2].

La **estructura de un pipe** es:

```
$ cmd1 | cmd2
```


**Ejemplo:** En un archivo se tiene un registro de las compras de una tienda, y deseamos saber el nombre de los clientes individuales y contar la cantidad de compras de cada uno, para poder desarrollar una estrategia de mercadeo.

Los pipes nos permiten realizar toda esta acción sin la necesidad de realizar un script adicional:

```
$ cat registro.txt | cut -d " " -f 1 | sort -n |  uniq -c
```

1. El comando `cat` recibe como entrada el archivo y su salida es el contenido del archivo.
2. El comando `cut` recibe el contenido del archivo como entrada y en su salida estará solamente la primera columna, es decir los nombres.
3. El comando `sort` recibe los nombres de los clientes y los agrupa por nombre.
4. El comando `uniq` cuenta la cantidad de veces que aparece cada nombre, luego elimina los elementos repetidos y presenta en su salida el nombre del cliente y la cantidad de compras que realizó.

**Pregunta 2.1**
Mediante el uso únicamente de línea de comandos genere un archivo llamado Ejercicio1.txt que contenga la siguiente información:

- ➔ Encabezado:
  - ◆ Nombre completo
  - ◆ Carné
- ➔ Solución del ejercicio:
  - ◆ Fecha actual del sistema
  - ◆ Usuario: nombre de usuario en el sistema, no debe escribirlo explícitamente
  - ◆ Versión del kernel de Linux
  - ◆ Lista de procesos del usuario actual

**a)** Comandos necesarios para generar el archivo descrito anteriormente:

```
❯ touch Ejercicio1.txt
❯ echo "Nombre: Esteban Alvarado Vargas" >> Ejercicio1.txt
❯ echo "Carne: 2018109336" >> Ejercicio1.txt
❯ echo "\nFecha: $(date)" >> Ejercicio1.txt
❯ echo "Usuario: $USER" >> Ejercicio1.txt
❯ echo "Kernel: $(uname -r)" >> Ejercicio1.txt
❯ echo "Lista de procesos:" >> Ejercicio1.txt
❯ ps >> Ejercicio1.txt
```

**b)** Contenido final del archivo:

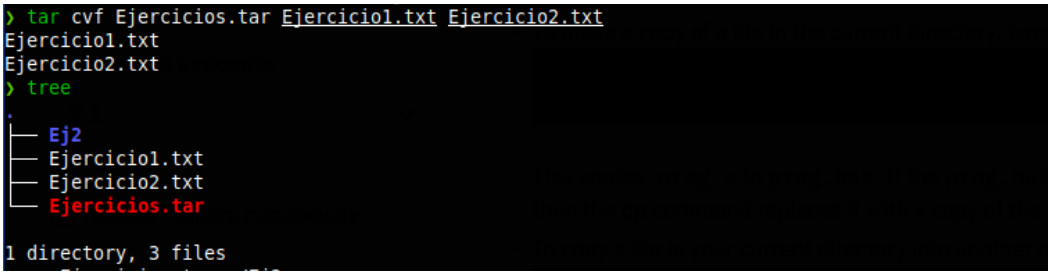
```
❯ cat Ejercicio1.txt
Nombre: Esteban Alvarado Vargas
Carne: 2018109336

Fecha: jue 10 feb 15:44:06 CST 2022
Usuario: estalvgs
Kernel: 5.4.0-99-generic
Lista de procesos:
    PID TTY          TIME CMD
  45379 pts/2    00:00:09 zsh
  45383 pts/2    00:00:00 zsh
  45846 pts/2    00:00:00 zsh
  45847 pts/2    00:00:00 zsh
  45849 pts/2    00:00:00 gitstatusd-linu
  48541 pts/2    00:00:00 ps
```

**Pregunta 2.2**
Dentro del directorio, utilice el `mkdir` para crear un nuevo directorio llamado `Ej2`. Muestre la rutina de comandos que se deben utilizar para realizar las siguientes operaciones:

- Generar un nuevo archivo `Ejercicio2.txt` que contenga la información de **`checksumMD5`** del archivo `Ejercicio1.txt`

```
❯ mkdir Ej2
❯ ls
Ej2  Ejercicio1.txt
❯ touch Ejercicio2.txt
❯ ls
Ej2  Ejercicio1.txt  Ejercicio2.txt
❯ md5sum Ejercicio1.txt >> Ejercicio2.txt
❯ cat Ejercicio2.txt
63e52d73e134cacfb37853821888a838  Ejercicio1.txt
```

- Realizar un archivamiento `Ejercicio.tar` a partir de ambos archivos: `Ejercicio1.txt` y `Ejercicio2.txt`.



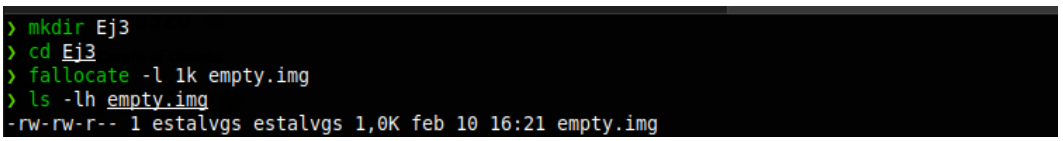

- Copiar el archivo `Ejercicio.tar` en el directorio recién creado `Ej2`.


- Ingresar al directorio `Ej2` y extraer allí el archivo `Ejercicios.tar`.

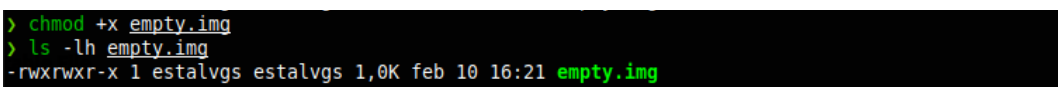

**Pregunta 2.3**

Dentro de un directorio llamado `Ej3`. Muestre la rutina de comandos que se deben utilizar para realizar las siguientes operaciones:

- Crear una imagen de archivo en blanco `empty.img`, con un tamaño de 1 kB.




- Cambiar permisos de archivo para convertirlo en ejecutable (permiso de ejecución).



- Escribir (por comandos) al archivo `empty.img` las siguientes líneas:

```
“clear”
“wget
http://www.bolis.com/onyx/random/stuff/sounds/murray/murrays.wav"
"aplay murrays.wav"
"mv murrays.wav .murrays.wav"
"echo Aqui no ha pasado nada"
```

```
❯ echo "clear" >> empty.img
❯ echo "wget http://www.bolis.com/onyx/random/stuff/sounds/murray/murrays.wav" >> empty.img
❯ echo "aplay murrays.wav" >> empty.img
❯ echo "mv murrays.wav .murrays.wav" >> empty.img
❯ echo "echo Aqui no ha pasado nada" >> empty.img
```

- Renombrar el archivo `empty.img` a `script.x`

```
❯ mv empty.img script.x
❯ tree
.
└── script.x

0 directories, 1 file
```

- Ejecutar `script.x` (¿Qué debe suceder en este punto?)

```
--2022-02-10 16:24:20--  http://www.bolis.com/onyx/random/stuff/sounds/murray/murrays.wav
Resolviendo www.bolis.com (www.bolis.com)... 173.236.153.129
Conectando con www.bolis.com (www.bolis.com)[173.236.153.129]:80... conectado.
Petición HTTP enviada, esperando respuesta... 200 OK
Longitud: 113452 (111K) [audio/x-wav]
Guardando como: “murrays.wav”

murrays.wav                    100%[=========================================================>] 110,79K   470KB/s

2022-02-10 16:24:20 (470 KB/s) - “murrays.wav” guardado [113452/113452]

Sonando WAVE 'murrays.wav' : Signed 16 bit Little Endian, Ratio 22050 Hz, Mono
Aqui no ha pasado nada
```

Creamos un archivo ejecutable con comandos que limpian la terminal, descarga un archivo de audio, lo reproduce, lo renombra (como archivo oculto) y al final imprime un mensaje.

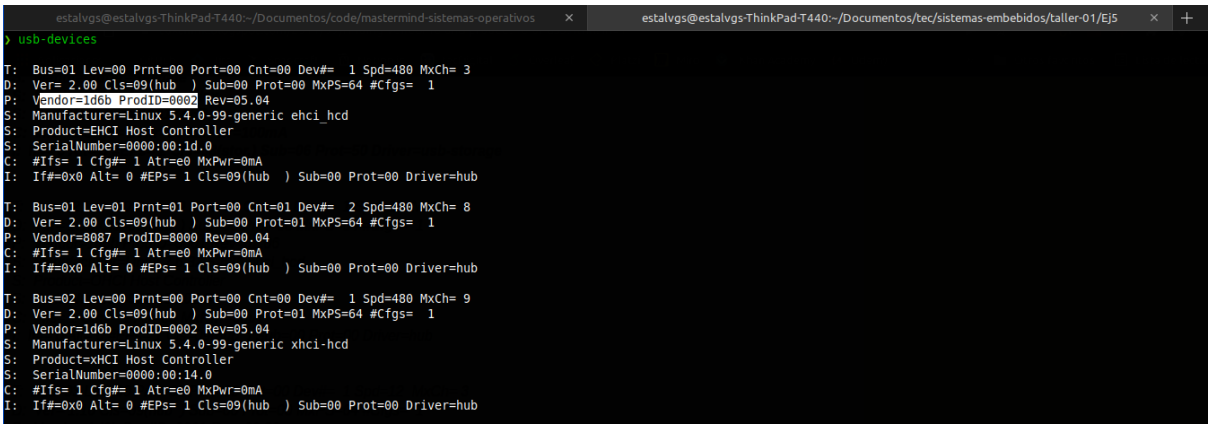
**Pregunta 2.4**

Muestre los comandos necesarios para descargar el archivo y luego mostrar en consola la cuenta de las líneas que incluyan la palabra *Zeus*.

```
❯ wget https://archive.org/stream/laodisea00homeuoft/laodisea00homeuoft_djvu.txt
--2022-02-10 16:41:35--  https://archive.org/stream/laodisea00homeuoft/laodisea00homeuoft_djvu.txt
Resolviendo archive.org (archive.org)... 207.241.224.2
Conectando con archive.org (archive.org)[207.241.224.2]:443... conectado.
Petición HTTP enviada, esperando respuesta... 200 OK
Longitud: no especificado [text/html]
Guardando como: “laodisea00homeuoft_djvu.txt”

laodisea00homeuoft_djvu.txt          [       <=>                                          ] 814,09K   437KB/s    en 1,9s

2022-02-10 16:41:38 (437 KB/s) - “laodisea00homeuoft_djvu.txt” guardado [833629]

❯ ls
laodisea00homeuoft_djvu.txt
❯ less laodisea00homeuoft_djvu.txt | grep -c "Zeus"
256
  ~/Do/tec/sistemas-embebidos/taller-01/Ej4 ❯                                    base  16:44:35
```

**Pregunta 2.5**

Dentro de un directorio llamado `Ej5`. Muestre un comando que pueda usar para saber el número de identificador de *Vendor* e identificador de producto de los dispositivos USB que han sido conectados al sistema recientemente, desde que inició.




---

**Referencias**

[1]R. Baig Viñas and F. Aulí Llinàs, *Sistema operativo GNU/Linux básico*. Barcelona: Fundació per a la Universitat Oberta de Catalunya, 2003.

[2]S. Allende, F. Gibellini, C. Sánchez and M. Serna, Sistemas operativos: Linux teoría y práctica, 2nd ed. Ciudad Autónoma de Buenos Aires: edUTecNe, 2019.

[3]R. Petersen, *Linux*, 6th ed. New York: McGraw-Hill, 2008.

[4]R. Blum and D. LeBlanc, Linux for dummies, 9th ed. Hoboken, N.J.: Wiley, 2009.