Los comandos básicos de Linux

Durante esta actividad practicaremos algunos de los comandos básicos de Linux que nos permiten gestionar un sistema, actividades como listar, copiar, mover, crear directorios y archivos harán parte de esta práctica.

El Comando pwd

`pwd` es un comando que permite identificar en que lugar del sistema te encuentras ubicado, es decir, en que carpeta estás actualmente

- Se ejecuta simplemente `pwd` para que te imprima en pantalla una ruta de donde te encuentras por ejemplo `/home/${USER}/directorio1/`

La siguiente ejecución te permitirá confirmar que estás en el directorio `/home/${USER}/`, por favor ubicate sobre la proxima ejecución y presiona en el botón "Run" para validarlo.

In [1]:

```
pwd
```

```
/home/jovyan
```

Al ejecutarlo podrás ver en pantalla una salida "Out" que te mostrará en que directorio realizaremos nuestra ejecución, por ejemplo `/home/jovyan/`, desde este directorio comenzaremos nuestra práctica.

El Comando mkdir

`mkdir` (abreviación de "make directory") es un comando que permite crear directorios en Linux

- Puedes crear directorios en la ruta actual con `mkdir directorioacrear` o en otras rutas `mkdir /ruta/al/directorio/directorioacrear`
- Puedes añadir flags como `-p` para crear varios directorios al tiempo `mkdir -p directorio1 directorio2 directorio3`
- Puedes anidar directorios utilizando corchetes `mkdir -p directorio1/{subdirectorio1,subdirectorio2}`

La siguiente ejecución creará el directorio `directorio1` y el directorio `directorio2` en la ruta absoluta `/home/${USER}/` (reemplaza ${USER}, por el nombre del directorio de usuario que viste en la ejecución anterior para que quede algo como `/home/jovyan/)`. Es posible que no veas nada en pantalla pero con el comando ls validaremos luego que todo salió bien.

In [2]:

```
mkdir -p /home/${USER}/directorio1 /home/${USER}/directorio2
```

La siguiente ejecución creará el directorio `documentos` y dentro de él los subdirectorios `universidad`, `trabajo` y `personales`, además dentro del directorio `universidad` crearemos los directorios `SEM1` y `SEM2`. Esta vez asumiremos que estamos en la ruta /home/${USER}/ por lo que utilizaremos una ruta relativa y no una ruta absoluta. Es posible que no veas nada en pantalla pero con el comando ls validaremos luego que todo salió bien.

In [3]:

```
mkdir -p documentos/{trabajo,personales,universidad/{SEM1,SEM2}}
```

El Comando ls

`ls` (abreviación de "list") es un comando que sirve para listar los archivos o directorios de un directorio

- Puedes añadir flags como "-la" para ver más información de los archvios por ejemplo `ls -la directorio`.
- Puedes añadir rutas como por ejemplo `ls /directorio/subdirectorio` para ver el contenido de otro directorio diferente al que estás usando actualmente

La siguiente ejecución listara el contenido del directorio actual en el que estás ubicado `/home/${USER}/` por lo que se deberán ver los directorios `directorio1`, `directorio2` y `documentos` creados en la anterior ejecución.

In [4]:

```
ls
```

```
archivodeaudio.mp3  directorios-importantes  unidad_1.ipynb  unidad_6.ipynb
binder              documentos               unidad_2.ipynb
directorio1         informacion-personal     unidad_4.ipynb
directorio2         README.md                unidad_5.ipynb
```

Ahora, en esta ejecución deberás listar el contenido del directorio documentos `/home/${USER}/documentos` pero utilizando una ruta absoluta y un flag para ver más información como permisos y propietarios `-la` allí se deberán ver los directorios `trabajo`, `personales` y `universidad` creados en la anterior ejecución. (reemplaza ${USER}, por el nombre del directorio de usuario que hemos estado utilizando en anteriores ejecuciones para que quede algo como `/home/jovyan/)`.

In [5]:

```
ls -la /home/${USER}/documentos/
```

```
total 24
drwxr-xr-x 5 jovyan jovyan 4096 Jul 29 03:01 .
drwxr-xr-x 1 jovyan jovyan 4096 Jul 29 03:01 ..
drwxr-xr-x 2 jovyan jovyan 4096 Jul 29 03:01 personales
drwxr-xr-x 2 jovyan jovyan 4096 Jul 29 03:01 trabajo
drwxr-xr-x 4 jovyan jovyan 4096 Jul 29 03:01 universidad
```

El Comando touch

`touch` es un comando que permite crear archivos vacíos en Linux

- Puedes crear archivos en la ruta actual con `touch archivoacrear` o en otras rutas `touch /ruta/al/directorio/archivoacrear`
- Puedes crear multiples archivos al tiempo `touch archivoacrear1 archivoacrear2 archivoacrear3`

La siguiente ejecución creara `archivo1.txt` en el `directorio1` y el archivo `archivo2.txt` en el `directorio2` en la ruta `/home/${USER}/`.

In [6]:

```
touch /home/${USER}/directorio1/archivo1.txt /home/${USER}/directorio2/archivo2.txt
```

Puedes validar que los archivos se crearon correctamente utilizando el comando `ls`.

In [7]:

```
ls /home/${USER}/directorio1/
```

```
archivo1.txt  archivodeaudio.mp3
```

In [8]:

```
ls /home/${USER}/directorio2/
```

```
archivo2.txt
```

El Comando mv

`mv` (abreviación de "move") es un comando que sirve para mover archivos y/o directorios de un lugar a otro

- Permite mover directorios de forma recursiva.
- Se utiliza el comando `mv /recurso/de/origen /directorio/de/destino`

La siguiente ejecución movera el archivo `archivo1.txt` desde el directorio `directorio1` al directorio `directorio2`.

In [9]:

```
mv /home/${USER}/directorio1/archivo1.txt /home/${USER}/directorio2/
```

Puedes validar que los archivos se movieron correctamente utilizando el comando `ls`.

In [10]:

```
ls /home/${USER}/directorio1/
```

```
archivodeaudio.mp3
```

In [11]:

```
ls /home/${USER}/directorio2/
```

```
archivo1.txt  archivo2.txt
```

La siguiente ejecución movera el directorio `documentos` y todos sus subdirectorios a un nuevo directorio llamado `directorios-importantes`.

In [12]:

```
mv /home/${USER}/documentos /home/${USER}/directorios-importantes
```

Puedes validar que el directorio se movió correctamente utilizando el comando `ls`.

In [13]:

```
ls /home/${USER}/
```

```
archivodeaudio.mp3  directorio2              README.md       unidad_4.ipynb
binder              directorios-importantes  unidad_1.ipynb  unidad_5.ipynb
directorio1         informacion-personal     unidad_2.ipynb  unidad_6.ipynb
```

El Comando cp

`cp` (abreviación de "copy") es un comando que sirve para copiar archivos y/o directorios de un lugar a otro

- Permite archivos simples o múltiples archivos.
- Para copiar directorios enteros de forma recursiva se debe utilizar la flag `-R` así `cp -R /directorio/de/origen/con/muchosarchivos/ /directorio/de/destino/`.

La siguiente ejecución copia el archivo `archivo1.txt` desde el directorio `directorio2` de vuelta al directorio `directorio1`.

In [14]:

```
cp /home/${USER}/directorio2/archivo1.txt /home/${USER}/directorio1/
```

Puedes validar que los archivos se copiaron correctamente utilizando el comando `ls`.

In [15]:

```
ls /home/${USER}/directorio1/
```

```
archivo1.txt  archivodeaudio.mp3
```

In [16]:

```
ls /home/${USER}/directorio2/
```

```
archivo1.txt  archivo2.txt
```

La siguiente ejecución copia el directorio `directorios-importantes` y todos sus subdirectorios de manera recursiva `-R` a un nuevo directorio llamado `informacion-personal`.

In [17]:

```
cp -R /home/${USER}/directorios-importantes /home/${USER}/informacion-personal
```

Puedes validar que el directorio se copió correctamente utilizando el comando `ls`.

In [18]:

```
ls /home/${USER}/
```

```
archivodeaudio.mp3  directorio2              README.md       unidad_4.ipynb
binder              directorios-importantes  unidad_1.ipynb  unidad_5.ipynb
directorio1         informacion-personal     unidad_2.ipynb  unidad_6.ipynb
```

El Comando cd

`cd` (abreviación de "change directory") es un comando que sirve para moverte entre directorios

- Puedes moverte utilizando rutas absolutas o relativas por ejemplo si estás en `/home/${USER}/` y quieres moverte a `directorio1` puedes utilizar:
- Rutas absolutas: `cd /home/${USER}/directorio1/` (importante que siempre que uses rutas absolutas antepongas la raíz `/`)
- Rutas relativas: `cd directorio1` (esto debido a que ya estás dentro de `/home/${USER}/` asi que solo te moverás un espacio hacia adelante.
- Puedes utilizar atajos, por ejemplo:
- Moverte un directorio hacia atras: `cd ..`
- Moverte dos o más directorios hacia atrás: `cd ../../` (tantos ../ como directorios desees moverte).

La siguiente ejecución del comando `pwd` te permitirá determinar en que directorio estás.

In [19]:

```
pwd
```

```
/home/jovyan
```

Ahora la siguiente ejecución te permitirá moverte al directorio `/home/${USER}/directorio1/` (reemplaza `${USER}` por el nombre del usuario del directorio)

In [20]:

```
cd /home/${USER}/directorio1/
```

Puedes validar en que directorio estás ahora ejecutando de nuevo el comando `pwd`.

In [21]:

```
pwd
```

```
/home/jovyan/directorio1
```

Ahora la siguiente ejecución te permitirá con el atajo `../` al directorio `/home/${USER}/directorio2/`

In [22]:

```
cd ../directorio2/
```

Puedes validar de nuevo en que directorio estás ahora ejecutando de nuevo el comando `pwd`.

In [23]:

```
pwd
```

```
/home/jovyan/directorio2
```

El Comando rm o el Comando rmdir

`rm` (abreviación de "remove") o `rmdir` (abreviación de "remove directory" menos utilizado) son comandos eliminar archivos o directorios.

- Se ejecuta simplemente `rm` borrar un solo archivo `rm /ruta/al/archivo/a/borrar/`
- Se ejecuta `rmdir` para borrar directorios vacíos `rmdir /ruta/al/directorio/vacio/`
- Con `rm` se ejecuta la flag `-f` que permite omitir la confirmación (forzar el borrado)
- Con `rm` se ejecuta la flag `-r` para poder borrar directorios con contenido en su interior (recursivamente)
- Dados los ejemplos anteriores con `rm` podríamos borrar un directorio con contenido de manera recursiva y forzada así `rm -rf /ruta/al/directorio/con/contenido/a/borrar/`

La siguiente ejecución de `rmdir` intentará eliminar el contenido del directorio `directorio2` sin embargo fallará debido a que ese directorio no está vacío.

In [24]:

```
rmdir /home/${USER}/directorio2/
```

```
rmdir: failed to remove '/home/jovyan/directorio2/': Directory not empty
```

La siguiente ejecución de `rm` con el flag `-rf` sí te permitirá eliminar el directorio2 y todo su contenido `/home/${USER}/directorio2/`

In [25]:

```
rm -rf /home/${USER}/directorio2/
```

Puedes validar que el directorio fue borrado con el comando `ls`.

In [26]:

```
ls /home/${USER}/
```

```
archivodeaudio.mp3  directorios-importantes  unidad_1.ipynb  unidad_5.ipynb
binder              informacion-personal     unidad_2.ipynb  unidad_6.ipynb
directorio1         README.md                unidad_4.ipynb
```

El Comando curl y el Comando wget

`curl` (abreviación de "client for URL's") es una poderosa herramienta que te permite transferir datos utilizando diferentes protocolos. `wget` (abreviación de "world wide web get") es una herramienta que permite obtener recursos de internet.

- Se ejecuta `curl` y el sitio web objetivo para hacer una consulta simple de GET `curl https://google.com/`
- Se ejecuta `wget` y el recurso en línea objetivo para hacer una descarga de archivo `wget https://sitio.web/archivoadescargar.txt`
- Se puede añadir el flag `-O` a `wget` para definir en donde se almacenará el recurso descargado.

La siguiente ejecución te permitirá descargar con curl un archivo de mp3 de ejemplo de un sitio web, y guardarlo en la ruta `/home/${USER}/directorio1/` con el nombre `archivodeaudio.mp3`.

In [27]:

```
wget https://file-examples.com/wp-content/uploads/2017/11/file_example_MP3_700KB.mp3 -O /home/${USER}/directorio1/archivodeaudio.mp3
```

```
--2021-07-29 03:01:41--  https://file-examples.com/wp-content/uploads/2017/11/file_example_MP3_700KB.mp3
Resolving file-examples.com (file-examples.com)... 185.135.88.81
Connecting to file-examples.com (file-examples.com)|185.135.88.81|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1084 (1.1K) [text/html]
Saving to: ‘/home/jovyan/directorio1/archivodeaudio.mp3’

/home/jovyan/direct 100%[===================>]   1.06K  --.-KB/s    in 0s

2021-07-29 03:01:42 (42.2 MB/s) - ‘/home/jovyan/directorio1/archivodeaudio.mp3’ saved [1084/1084]
```

Puedes validar que el archivo fue creado con el comando `ls`.

In [28]:

```
ls /home/${USER}/directorio1/
```

```
archivo1.txt  archivodeaudio.mp3
```

El Comando ln

`ln` (abreviación de "link") crea enlaces simbólicos de archivos en Linux, los enlaces simbólicos son una especie de "accesos directos" (aunque no es la definición más acertada) de recursos en Linux, por ejemplo, binarios.

- Se utiliza `ln origen destino`
- Se utilizan diferentes flags como `-f` para forzar en caso de falla o `-b` para conservar el nombre original.

La siguiente ejecución te permitirá crear un enlace simbólico del archivo de audio `archivodeaudio` ubicado en `/home/${USER}/directorio1/` en el directorio `/home/${USER}/`.

In [29]:

```
ln /home/${USER}/directorio1/archivodeaudio.mp3 /home/${USER}/archivodeaudio.mp3
```

```
ln: failed to create hard link '/home/jovyan/archivodeaudio.mp3': File exists
```

Puedes validar que el enlace simbólico fue creado con el comando `ls`.

In [30]:

```
ls /home/${USER}
```

```
archivodeaudio.mp3  directorios-importantes  unidad_1.ipynb  unidad_5.ipynb
binder              informacion-personal     unidad_2.ipynb  unidad_6.ipynb
directorio1         README.md                unidad_4.ipynb
```

También es posible ejecutar el comando para crear dicho archivo con un nombre diferente:

In [31]:

```
ln /home/${USER}/directorio1/archivodeaudio.mp3 /home/${USER}/archivodeaudio-nuevo.mp3
```

Validamos el nuevo enlace simbólico.

In [32]:

```
ls /home/${USER}
```

```
archivodeaudio.mp3        directorios-importantes  unidad_2.ipynb
archivodeaudio-nuevo.mp3  informacion-personal     unidad_4.ipynb
binder                    README.md                unidad_5.ipynb
directorio1               unidad_1.ipynb           unidad_6.ipynb
```

Si ejecutas de nuevo el anterior comando para tratar de sobreescribir el enlace simbolico va a fallar. Probemos:

In [33]:

```
ln /home/${USER}/directorio1/archivodeaudio.mp3 /home/${USER}/archivodeaudio-nuevo.mp3
```

```
ln: failed to create hard link '/home/jovyan/archivodeaudio-nuevo.mp3': File exists
```

Sin embargo con la opción `-f` se forzará su ejecución.

In [34]:

```
ln -f /home/${USER}/directorio1/archivodeaudio.mp3 /home/${USER}/archivodeaudio-nuevo.mp3
```

El Comando find

`find` te permitirá buscar archivos y/o directorios.

- Se utiliza `find ruta opciones patron`.
- Se pueden utilizar wildcards por ejemplo *.txt permitirá buscar todos los archivos.txt
- Se pueden utilizar flags como `--name` para buscar el nombre exacto o `-iname` para buscar el nombre exacto sin importar si son mayúsculas o minúsculas.

La siguiente ejecución te permitira buscar todos los archivos `.mp3` del directorio `/home/${USER}/`

In [36]:

```
find /home/${USER}/ -iname "*.mp3"
```

```
/home/jovyan/directorio1/archivodeaudio.mp3
/home/jovyan/archivodeaudio.mp3
/home/jovyan/archivodeaudio-nuevo.mp3
```

Con esta deberías poder ver la ubicación de `archivodeaudio.mp3`

El Comando date

`date` te dará la hora y fecha del sistema. `wget` (abreviación de "world wide web get")es una herramienta que permite obtener recursos de internet.

- Puedes utilizar cientos de formatos diferentes algunos ejemplos son:
- `date +%Y-%m-%d` Te dará la fecha en este formato: 2020-08-02
- `date +%D` Te dará la fecha en este formato: 08/02/20
- `date --date="1 day ago"` Te dará la fecha y hora de un día atrás.

La siguiente ejecución te dará una salida simple del formato date

In [37]:

```
date
```

```
Thu Jul 29 03:10:52 UTC 2021
```

El Comando grep

`grep` te permitirá filtrar una palabra dentro de la ejecución de un comando o dentro de un archivo.

- Se utiliza `grep` opciones patron archivo.

(Veremos ejemplos de este comando en el siguiente ejercicio)

El Comando cat

`cat` te permitirá imprimir en pantalla el contenido de un archivo.

- Se utiliza `cat archivo.txt`

(Veremos ejemplos de este comando en el siguiente ejercicio)

Las redirecciones simples (>) y dobles (>>)

- Las redirecciones simples (>) te permitirán, por ejemplo, escribir la salida de un comando a la primera línea de un archivo, sobreescribiendo todo su contenido, incluso si este archivo no existe lo podrá crear, por ejemplo:

`comando > archivo.txt`

- Las redirecciones dobles harán algo similar, sin embargo, no sobreescribirán el archivo, sino que enviarán la salida del comando a la última línea del mismo.

`comando >> archivo.txt`

La siguiente ejecución reenviará la salida del comando `ls /home/${USER}/directorio1/` a el archivo `archivodeprueba.txt`

In [38]:

```
ls /home/${USER}/directorio1/ > /home/${USER}/archivodeprueba.txt
```

In [39]:

```
cat /home/${USER}/archivodeprueba.txt
```

```
archivo1.txt
archivodeaudio.mp3
```

In [40]:

```
date > /home/${USER}/archivodeprueba.txt
```

Puedes validar el contenido del archivo `archivodeprueba.txt` con el comando `cat`

In [41]:

```
cat /home/${USER}/archivodeprueba.txt
```

```
Thu Jul 29 03:14:41 UTC 2021
```

La siguiente ejecución reenviará la salida del comando `date --date="1 month ago"` (la fecha de hace un mes) a el archivo `archivodeprueba.txt`, pero lo hará en una nueva línea del archivo, sin sobreescribir su contenido.

In [42]:

```
date --date="1 month ago" >> /home/${USER}/archivodeprueba.txt
```

Puedes validar el contenido del archivo `archivodeprueba.txt` con el comando `cat`

In [43]:

```
cat /home/${USER}/archivodeprueba.txt
```

```
Thu Jul 29 03:14:41 UTC 2021
Tue Jun 29 03:14:56 UTC 2021
```

Las tuberías o pipe (|)

- Las tuberías sirven para enviar la salida de un comando a otro comando, por ejemplo, sabemos que con `cat` podemos ver el contenido del archivo `/home/${USER}/archivodeprueba.txt` que ahora tiene en su interior algo como: `dom 09 ago 2020 12:51:56 -05` `jue 09 jul 2020 12:54:58 -05`

Con el uso de la tubería `|` y el comando `grep` filtraremos la primera línea de ese archivo con el nombre del mes, por ejemplo:

`cat /home/${USER}/archivodeprueba.txt | grep "ago"

Haremos esta ejecución a continuación.

Imprimiremos con `cat` el contenido del archivo `/home/${USER}/archivodeprueba.txt`

In [44]:

```
cat /home/${USER}/archivodeprueba.txt
```

```
Thu Jul 29 03:14:41 UTC 2021
Tue Jun 29 03:14:56 UTC 2021
```

Queremos filtrar la primera línea de ese archivo, así que filtraremos con `grep` el mes de esa primera línea `/home/${USER}/archivodeprueba.txt` cambia el texto `${MES}` por el mes a filtrar (por ejemplo `grep "Oct"`).

In [48]:

```
cat /home/${USER}/archivodeprueba.txt | grep "Jul 29"
```

```
Thu Jul 29 03:14:41 UTC 2021
```

Deberás ver solo la línea filtrada en lugar de dos líneas.

Los ampersand (&)

- Los ampersand simples en Linux sirven para ejecutar multiples comandos en una única línea sin importar si alguno de los comandos falla o no.

`commando1 & commando2 & commando3`

- Los ampersand dobles no permiten continuar la secuencia de ejecución de los comandos, hasta que se ejecuten (uno por uno), de manera correcta.

`commando1 && commando2 && commando3`

La siguiente ejecución listará el contenido del `directorio1` y luego el contenido del `directorio2`

In [49]:

```
ls -la /home/${USER}/directorio1 & ls -la /home/${USER}/
```

```
[1] 171
total 16
drwxr-xr-x 2 jovyan jovyan 4096 Jul 29 03:01 .
drwxr-xr-x 1 jovyan jovyan 4096 Jul 29 03:18 ..
-rw-r--r-- 1 jovyan jovyan    0 Jul 29 03:01 archivo1.txt
-rw-r--r-- 3 jovyan jovyan 1084 Jul  9  2020 archivodeaudio.mp3
total 204
drwxr-xr-x 1 jovyan jovyan  4096 Jul 29 03:18 .
drwxr-xr-x 1 root   root    4096 May 14 19:51 ..
-rw-r--r-- 3 jovyan jovyan  1084 Jul  9  2020 archivodeaudio.mp3
-rw-r--r-- 3 jovyan jovyan  1084 Jul  9  2020 archivodeaudio-nuevo.mp3
-rw-r--r-- 1 jovyan jovyan    58 Jul 29 03:14 archivodeprueba.txt
-rw------- 1 jovyan jovyan  1378 Jul 29 03:01 .bash_history
-rw-r--r-- 1 jovyan jovyan   220 Apr  4  2018 .bash_logout
-rw-r--r-- 1 jovyan jovyan  3771 Apr  4  2018 .bashrc
drwxr-xr-x 1 jovyan jovyan  4096 Jun 22 00:48 binder
drwxr-xr-x 3 jovyan jovyan  4096 Jul 29 02:55 .cache
drwx------ 3 jovyan jovyan  4096 May 14 19:54 .config
drwxr-xr-x 2 jovyan jovyan  4096 Jul 29 03:01 directorio1
drwxr-xr-x 6 jovyan jovyan  4096 Jul 29 03:01 directorios-importantes
drwxr-xr-x 8 jovyan jovyan  4096 Jun 22 00:48 .git
drwxr-xr-x 6 jovyan jovyan  4096 Jul 29 03:01 informacion-personal
drwxr-xr-x 2 jovyan jovyan  4096 Jul 29 02:58 .ipynb_checkpoints
drwxr-xr-x 6 jovyan jovyan  4096 Jul 29 02:56 .ipython
-rw-r--r-- 1 jovyan jovyan  5278 Jul 29 03:18 .jupyter-server-log.txt
drwxr-xr-x 3 jovyan jovyan  4096 Jul 29 02:55 .local
-rw-r--r-- 1 jovyan jovyan   807 Apr  4  2018 .profile
-rw-r--r-- 1 jovyan jovyan   146 Jun 22 00:48 README.md
-rw-r--r-- 1 jovyan jovyan 36911 Jul 29 03:18 unidad_1.ipynb
-rw-r--r-- 1 jovyan jovyan 21230 Jun 22 00:48 unidad_2.ipynb
-rw-r--r-- 1 jovyan jovyan 13194 Jun 22 00:48 unidad_4.ipynb
-rw-r--r-- 1 jovyan jovyan  9889 Jun 22 00:48 unidad_5.ipynb
-rw-r--r-- 1 jovyan jovyan 12964 Jun 22 00:48 unidad_6.ipynb
[1]+  Done                    ls --color=auto -la /home/${USER}/directorio1
```