**Comandos Bash**

**pwd**: muestra el directorio actual en el que estamos

**clear**: limpia la terminal

**which bash**: nos dice que terminal tenemos

**ls**: lista los archivos en ese directorio

**ls -a**: lista todos los archivos en ese directorio incluyendo ocultos

**ls -la**: lista los elementos ocultos

**ls -l**: Lista archivos en formato largo (mostrando sus permisos, siendo la primera letra el tipo(**d** directorio, **l** links y **-** archivo), seguido de 3 letras(rwx) de permisos para dueño, 3 letras(rwx) de permisos para grupo del dueño y 3 letras(rwx) de permisos para usuarios ajenos)

**cd**: entrar a carpetas

cd ~: ir al directorio del usuario

cd -: ir a la ubicacion anterior

**;**: sirve para encadenar comandos sin importar si el anterior es valido por ejemplo(cd.. ; ls <- OR)

**&&**: sirve para encadenar comandos ejecuta el actual si el anterior fue valido(cd.. && ls <- AND)

**man (comando)**: Da informacion sobre la sintaxis y que hace el comando

**head (archivo)**: visualiza las primeras lineas de un archivo

**head -n** (numero de linea) (archivo): visualiza la cantidad de lineas que queres ver de un archivo

**tail (archivo):** visualiza las ultimas lineas de un archivo

**tail -n (numero de linea) (archivo)**: visualiza la cantidad de lineas que queres ver de un archivo (empezando de abajo)

**less (archivo)**: visualizar archivo en VIM

**touch (archivo)**: cambia la fecha de un archivo / crea archivos vacios

**nano (archivo)**: crear archivo y poner datos

**mkdir (carpeta)**: crea una carpeta nueva

**mkdir -p carpeta1/carpeta2/carpeta3**: permite crear varias carpetas una adentro de la otra

**rm (archivo/carpeta)**: elimina el archivo o la carpeta especificada

**rm -i (archivo/carpeta)**:pide confirmacion antes de eliminar y/n

**rmdir (carpeta)**: elimina una carpeta vacia

**rm -rf (carpeta)**: elimina toda una carpeta

**tree**: muestra el path de una forma mas comoda

**mv (archivo a mover) (path)**: mover archivos entre direcciones

**mv (archivo a renombrar) (nuevo nombre)**: renombrarlos

**cp (archivo a copiar) (path)**: crea una copia en la direccion asignada

**grep (patrón a buscar) (archivo)**: muestra la cantidad de veces que aparece ese patron

**grep -c (patrón a buscar) (archivo)**: muestra el numero de ocurrencias de ese patron

**grep -l (patrón a buscar) (archivo1) (archivo2) (archivo3)**: muestra en la cantidad de archivos en los que aparece la palabra

**grep -c (patrón a buscar) (archivo1) (archivo2) (archivo3)**: muestra el numero de ocurrencias de la palabra dentro de los archivos

**grep -c (patron a buscar) \* (path/\*)**: muestra TODAS las apariciones del patron dentro del path

**grep -n (patron a buscar) \* (path/\*)**: muestra las lineas en las que aparece dentro del path

**>**: operador de redireccion, el output que recibamos antes de este operador los va a mandar a un archivo (ls > output.txt (envia el output de ls a output.txt))

cat: concatenar o mostrar archivos

**cat > (archivo1)**: puedo escribir dentro del archivo

**cat (archivo1)**: muestra el archivo

**cat (archivo1) (archivo2)**: muestra en conjunto los archivos

**cat (archivo1) > (archivo2)**: SI NO EXISTE EL ARCHIVO puedo copiar informacion de archivo1, crear archivo2 y pegarlo ahi

**cat (archivo1) > (archivo2)**: SI YA EXISTE SOBREEESCRIBE EL ARCHIVO

**cat (archivo1) (archivo2) > (archivo3)**: combino el texto de los dos archivos dentro del archivo3

**//formas de abrir un archivo con espacios**

**cat “nombre con espacios”**

**cat nombre\ con\ espacios**

**cat no\***

**cat \*ame**

**cat \*this\***

**//formas de modificar los grupos/dueño de un archivo**

**sudo chown root:root (archivo)**: le pones al archivo como dueño al root y como grupo al root ,el grupo es opcional.

**//formas de modificar los permisos de un archivo**

**sudo chmod 700 (archivo)**: le pones al archivo todos los permisos al dueño, 0 a lo pertenecientes al grupo y 0 a los demas usuarios

-R sirve para hacer recursiva el cambio de los permisos y es un directorio

Lectura R **4**

Escritura W **2**

Ejecución X **1**

El comando **at** nos permite ejecutar un comando o script (o varios comandos o scripts) una única vez a una fecha y hora indicada

**dmesg** (diagnostic message, mensajes de diagnóstico) es una instrucción presente en los sistemas operativos Unix que lista el buffer de mensajes del núcleo.

El comando **locate** es la forma más práctica de buscar un archivo en Linux, este comando locate se encarga de realizar las búsquedas haciendo uso de una base

de datos donde se encuentra toda la información y locate la consulta a ella directamente.

locate algo

Cuando decimos que el comando **find** es más útil y con mejores resultados que locate se debe a razones como:

Puede buscar archivos asociados a un determinado usuario o grupo de usuarios.

Puede buscar archivos que fueron modificados o accedidos recientemente.

Permite la búsqueda de archivos con un rango de tamaño específico, archivos ocultos, ejecutables, archivos de solo lectura y archivos con permisos específicos.

Pero es más lento busca en cada directorio, en cada partición ,etc

**sudo find -name** “\*Solvetic\*”: busca en donde estoy parado cualquier nombre q tenga Solvetic

o podes hacer find -name "\*.txt" busca los que terminen con .txt

**find -name "ar\*"** busca los que empiecen con ar

**find /ruta -name nombreDeAlgo**

**find .**: empiezo a buscar desde el directorio actual

**find ~** empiezo a buscar desde mi directorio de usuario

**useradd [options] nombre\_de\_usuario**

useradd nuevoUsuario: Crea el usuario nuevoUsuario pelado, sin contraseña, sin interprete de comandos, sin directorio de usuario.

**useradd -m -d /home/depruebas -s /bin/bash depruebas** lo crea asignandole un home , una shell pero sin contraseña.

Ejemplo, si deseamos crear un usuario llamado 'pedro' cuyo grupo principal sea 'profesores', cuya carpeta home sea /home/pedro y su intérprete de comandos sea /bin/bash, ejecutaremos el siguiente comando:

**//Crear un usuario**

**sudo useradd -g profesores -d /home/pedro -m -s /bin/bash pedro**

los cambios se ven en el /etc/passwd y el /etc/group

agregar grupo a etc/group con $ sudo groupadd nombreGrupo

con usermod podes modificar cosas de los users , por ejemplo el nombre:

**$ sudo usermod -l nombre\_nuevo nombre\_antiguo**

o cambiar la contra de un user $ sudo usermod -p contraseña nombre\_usuario.El usermod -p espera que los datos sean la contraseña ya encriptada no se recomienda hacerlo de esta manera

La manera de cambiar la contraseña es con $ sudo passwd user

Agregar un usuario existente a un grupo existente

Si el usuario ya existe y el grupo tambien usamos:

**usermod -a -G contabilidad pedro**

Cambiar un usuario de grupo

Supongamos que no queremos que pedro pertenezca al grupo usuarios, y que este en el grupo retirados.

**usermod -g retirados pedro**

**//Calcular espacio ocupado por un archivo/directorio**

**du -sh (directorio/archivo)**

-s Suma todos tamaños de los elementos y muestra el total

-h lista los tamaños de un directorio y detalla el tamaño del archivo (KB, MB, GB…)

**du -hc (directorio/archivo)**

-h lista los tamaños de un directorio y detalla el tamaño del archivo (KB, MB, GB…)

-c lista todos los archivos y su tamaño sin tamaño pero con la suma total

**chgrp** sirve para cambiar el grupo de un archivo o directorio

**chgrp desarrollo algo.txt**  algo.txt pasa a pertenecer al grupo desarrollo

*//\*\*Comprimir varios archivos en uno solo\*\*//*

**tar**:El comando TAR es usado para comprimir o

empaquetar un archivo o directorio.

**ara crear una compresión tar, use la sintaxis del comando:**

$ tar -cvf /ruta/archivo.tar /ruta/origen

**Para crear un archivo comprimido gzip, usamos la opción z.**

La sintaxis del comando para crear un archivo .gz con tar es la siguiente:

$ tar -czvf /ruta/archivo.tgz /ruta/origen

**Para extraer un archivo que a menudo tiene una extensión .tar**, la sintaxis del comando es:

$ tar -xvf /ruta/archivo.tar -C /directorio/donde/quiero/dejarLaExtraccion

**Para extraer un archivo .gz**, usamos la sintaxis de comando:

$ tar -xzvf /ruta/archivo.tgz

**gzip:**

El comando Gzip crea un fichero comprimido que finaliza con .gz; Gunzip extrae los ficheros comprimidos y suprime el fichero .gz

**Para comprimir :**

gzip filename.ext

El fichero será comprimido y guardado como filename.ext.gz

**Y para descomprimirlo :**

gunzip filename.ext.gz

El filename.ext.gz se borra y se reemplaza con filename.ext.

***//\*\*\*fin de las cosas comprimidas\*\*//***

**El comando zgrep** se usa para buscar expresiones de un archivo dado, incluso si está comprimido. Todas las opciones

que se aplican al comando grep también se aplican al comando zgrep .

zgrep [grep options] cosaAbuscar archivo/archivos

**El comando wc** (recuento de palabras) en los sistemas operativos Unix / Linux se usa para averiguar el número de recuento de nuevas líneas,

Recuento de palabras, bytes y recuento de caracteres en un archivo especificado por los argumentos del archivo. La sintaxis del comando wc como se muestra a continuación.

**parámetros**:

-c muestra cantidad de bytes.

-l cantidad de líneas

-w --words muestran sólo palabras.

-m cantidad de caracteres

En caso de no pasar parametro wc se muestra cantidad de líneas,cantidad de palabras del archivo , cantidad de bytes que tiene el archivo. Comando es:

wc testfile

Comando ps , muestra procesos activos

**rev (algo)** da vuelta ese algo echo hola | rev (muestra aloh)

**diff** te muestra la diferencia entre archivos o directorio :

**diff -rq /home /home/iso** te va a detallar las diferencias que hay entre los directorios.

**diff -rq algo.txt sarasa.txt** solo te va a decir si son iguales o no , sin detallar porque no puede.

**awk:**

Los usos básicos que podemos dar al comando Awk son los siguientes:

1. **Buscar palabras y patrones de palabras y reemplazarlos por otras palabras y/o patrones.**
2. **Hacer operaciones matemáticas.**
3. **Procesar texto y mostrar las líneas y columnas que cumplen con determinadas condiciones.**

**file (archivo):** sirve para saber que tipo de archivo es “archivo”

**file \***: sirve para saber del directorio en donde estas parado todos los tipos de archivos

**usermod root −d /home/newroot −L**: ahora el nuevo directorio home de root es newroot y además(independientemente a lo anterior) con -L bloquea al usuario root.

**tty**: nos dice que numero de terminal somos

**mesg**: sirve para habilitar o deshabilitar la recepción de mensajes

**y**: habilitada la recepción de mensajes

**n**: deshabilitada la recepción de mensajes

**write (nombre del usuario) (ttx)**:

Se abre un panel para escribir mensajes (Ctrl+d para terminar de escribir)

**//Cambiar de usuario (terminal y gráfica)**

**Ctrl+F1** / **Ctrl+F2** / **Ctrl+F3**: entornos gráficos

**Ctrl+F4** / **Ctrl+F5**/**Ctrl+F…**/ **Ctrl+F12**: terminales

**//Ver información del equipo**

**hostnamectl**: muestra información del equipo

**hostnamectl set-hostname (nombrenuevo)**: cambia el nombre de la computadora.

**nice** ejecuta un comando con una prioridad determinada, o modifica la prioridad a de un proceso (programa en ejecución). Utiliza una prioridad variable que parte de la prioridad del shell y suma o resta valores. Mientras menor es el valor de la prioridad mayor prioridad tiene el proceso.

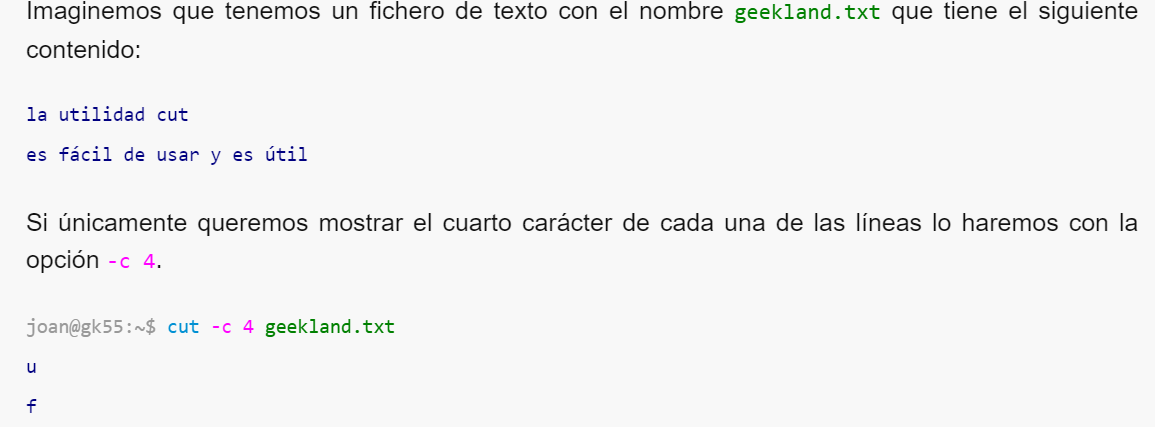
Su sintaxis es de la forma:

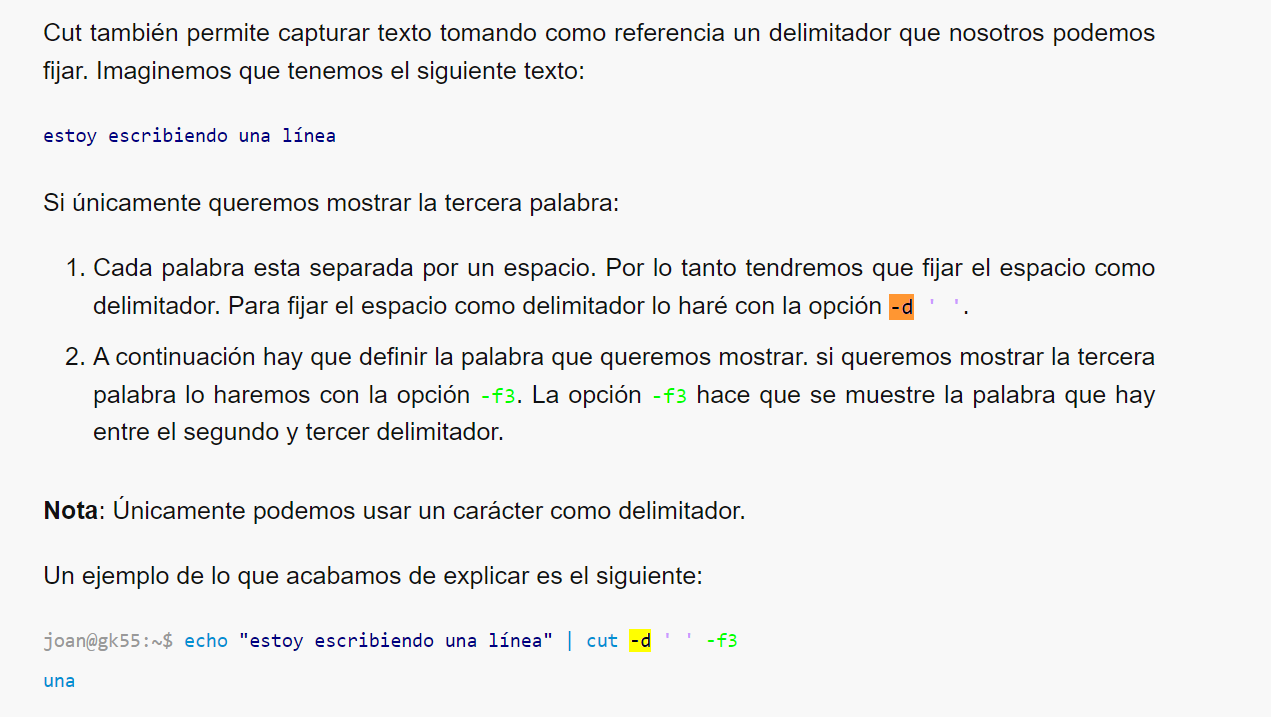
nice -n prioridad [comando]

sudo nice -n -5 find El valor de la prioridad del proceso find decrece en 5, aumentando su prioridad nice -n +5 find El valor de la prioridad del proceso find aumenta en 5, disminuye su prioridad.

comando **cut**

Su principal utilidad es la de borrar secciones, campos o caracteres de la salida de un comando o de cada una de las líneas de un fichero de texto.





Buscar mas info sobre **xargs**, va combinado con |

Cuando queremos usar el output de un comando como argumento de otro comando usar xargs o $().

Es lo mismo hacer

**find . -name arch\_uno | xargs cat**

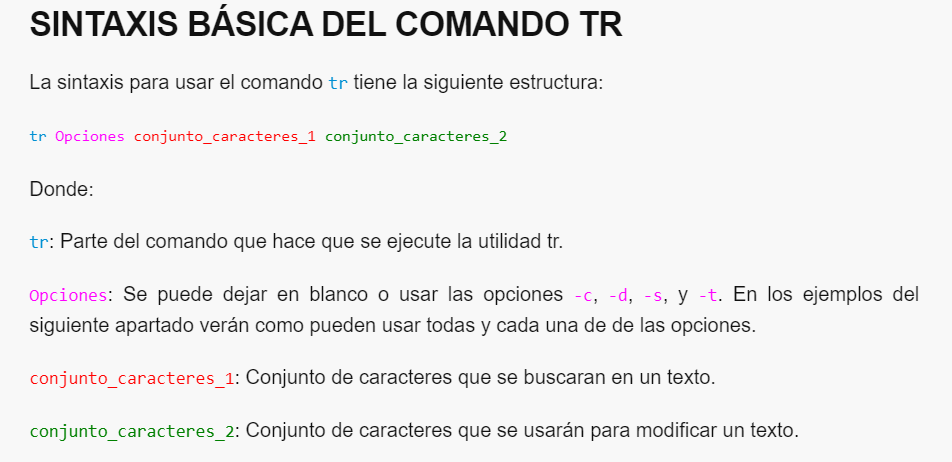
que hacer

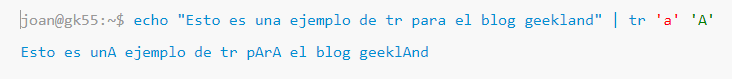
**cat $(find . -name arch\_uno)**

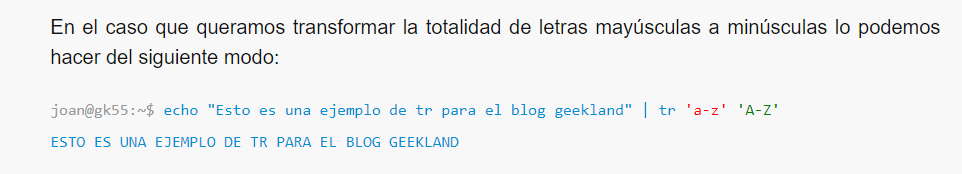
Si estoy parado en una carpeta en especifico, se puede hacer **ls | xargs chmod 400**, y se le cambia a todos los elementos que listó el ls los permisos

find / -size 1033c !-exec -type f -readable

**tr** sirve para manipular texto









**Comandos Shell Scripting**

Para crear variables se usa:

VARIABLE=3 , sin espacios alrededor del “=”.

Y se accede **echo $VARIABLE.**

---------------------------------------------------------------------------------

**nombre=Carlos**

**echo "Hola $nombre"** # Hola Carlos

**echo Hola ${nombre}** # Hola Carlos

**nombre=5**

**echo "Hola $nombre" # Hola 5**

**//Los arreglos**

**arreglo\_a=()** # Se crea vacıo

**arreglo\_b=(1 2 3 5 8 13 21)** # Inicializado

**//Asignar valor a posición en arreglo:**

**arreglo\_b[2]=spam**

**//Acceso al arreglo:**

**echo ${arreglo\_b[2]}**

**copia=${arreglo\_b[2]}**

para acceder a todas las posiciones del arreglo es : **echo ${arreglo\_b[@ o \*]}**

**//TEST para Estructuras de control**

Expresión Valor **True**

test ! expresión La expresión es falsa

test -n String El tamaño del String es mayor a 0

test -z String El tamaño del String es 0

test String1 = String2 Los Strings son iguales

test String1 != String2 Los Strings son distintos

test Int1 -eq Int2 Int1 es igual a Int2

test Int1 -gt Int2 Int1 es mayor a Int2

test Int1 -lt Int2 Int1 es menor a Int2

test -d ARCHIVO Existe ARCHIVO y es un directorio

test -e ARCHIVO Existe ARCHIVO

test -r ARCHIVO Existe ARCHIVO y tiene permiso de lectura

test -w ARCHIVO Existe ARCHIVO y tiene permiso de escritura

test -x ARCHIVO Existe ARCHIVO y tiene permiso de ejecución

test -s ARCHIVO Existe ARCHIVO y no está vacío

**//Estructuras de control**

* **if**

**if** [ <test> ]

**then**

<bloque if>

**elif** [ <test> ]

**then**

<bloque elif>

**else**

<bloque else>

**fi**

* **case**

**case** <variable> **in**

<opcion1>)

<bloque opcion1>

;;

<opcion2>)

<bloque opcion2>

;;

**esac**

* **while**

**while** [test]

**do**

<bloque while>

**done**

* **until**

**until** [test]

**do**

<bloque until>

**done**

* **for**

**for** ((INICIO;FINAL; INCREMENTO))

**do**

<bloque for>

**done**

* **select**

**select** ITEM **in** [LIST]

**do**

<bloque select>

**done**

**//Funciones y Parámetros**

//Declaración de la función

nombreFuncion(){

<bloque de la función>

}

//Llamado a la función

nombreFuncion $par1 $par2 par3 ⇐ Parámetros siendo par1 y par2 variables y par3 string

//Recepción de parámetros

$1…#N ⇐ Forma de acceder a los parámetros enviados a la función

$0 ⇐ Nombre del archivo

$\* ⇐ Muestra **TODOS** los argumentos

$# ⇐ Muestra la cantidad de argumentos