Bash:

A veces hay códigos o procesos o resultados a los que queremos llegar a lo que nos es más fácil llegar mediante el uno del prompt o terminal del sistema, en este caso aprenderemos como usar el lenguaje bash, del OS Linux.

Pipelines:

Se le llama pipelines al hecho hacer la entrada de un archivo, proceso, script, programa, metodo o función, la salida de otro, para esto se usa el carácter pipe |. Este carácter se puede usar tantas veces como se necesite para hacer las modificaciones necesarias del primer input para que llegue al output que necesitamos.

En caso de necesitarse solamente un pipeline podemos usar el carácter < para entrada y el carácter > para salida, el uno de estos sobrescribirá el archivo o creará uno nuevo en caso de que este no exista, como con el uso de “w” como permiso, para poder simplemente añadir más contenido al archivo como con el uso del permiso “a” usamos el carácter >>. Estos se usan solamente para la corriente de salida y entrada normales, para la corriente de salida de errores, podemos usar el numero dos, que hace referencia a esta 2>.

Signals:

Las señales son datos que se le envían a un proceso para que este se detenga por medio de ^C, para que termine limpiamente, ^Z para que se detenga, o por medio de kill process usando una segunda terminal, lo que devolverá un texto diciendo que el proceso está terminated.

Variables:

Las variables en bash se deben de definir con un signo igual, sin comillas para definir el tipo de dato, y no puede haber espacios entre la variable, el signo de igual y su valor.

Globs:

Son caracteres especiales:

\*: Se refiere a todos los archivos del directorio.

\*.py: Se refiere a todos los archivos que terminen, porque está detrás del asterisco, con .py, por lo que devolverá todos los archivos de extensión .py que existen en el directorio.

c\*: Se refiere a todos los archivos que comiencen, porque está detrás del asterisco, con c, por lo que devolverá todos los archivos que comiencen con c que existen en el directorio.

???: Se refiere a todos los archivos que su len sea la cantidad de ? que tenga, también se puede conbinar con caracteres para poder obtener =, todos los archivos de esa long que contengan los caracteres que le pasemos: ????.sh, de3volvera todas las palabras de 4 letras que terminen con . sh, que es la ectension de bash.

**Trabajando con archivos y directorios:**

* **cd** directorio cambia al directorio que le hemos pasado o hace referencia al current directory
* **pwd:** imprime el current working directory
* **ls:** lists the contents of the current directory
* **ls** directory: lists the contents of the received directory
* **ls** -l: lists the additional information for the contents of the directory
* **ls** -a: lists all files, including those hidden
* **ls** -la: applies both the -l and the -a flags
* **mkdir** directory: creates the directory with the received name
* **rmdir** directory: deletes the directory with the received name (if empty)
* **cp** old\_name new\_name: copies old\_name into new\_name
* **mv** old\_name new\_name: moves old\_name into new\_name
* **touch** file\_name: creates an empty file or updates the modified time if it exists
* **chmod** modifiers files: changes the permissions for the files according to the provided modifiers; we've seen +x to make the file executable
* **chown** user files: changes the owner of the files to the given user
* **chgrp** group files: changes the group of the files to the given group

**Operating with the content of files**

* **cat** file: shows the content of the file through standard output
* **wc** file: counts the amount of characters, words, and lines in the given file; can also count the same values of whatever it receives via stdin
* **file** file: prints the type of the given file, as recognized by the operating system
* **head** file: shows the first 10 lines of the given file
* **tail** file: shows the last 10 lines of the given file
* **less** file: scrolls through the contents of the given file (press "q" to quit)
* **sort** file: sorts the lines of the file alphabetically
* **cut -d**separator **-f**fields file: for each line in the given file, splits the line according to the given separator and prints the given fields (starting from 1)

**Additional commands**

* **echo** "message": prints the message to standard output
* **date:** prints the current date
* **who:** prints the list of users currently logged into the computer
* **man** command: shows the manual page of the given command; manual pages contain a lot of information explaining how to use each command (press "q" to quit)
* **uptime:** shows how long the computer has been running
* **free:** shows the amount of unused memory on the current system

### Operating with processes

These are some commands that are useful to know in Linux when interacting with processes. Not all of them are explained in videos, so feel free to investigate them on your own.

* **ps:** lists the processes executing in the current terminal for the current user
* **ps** ax: lists all processes currently executing for all users
* **ps** e: shows the environment for the processes listed
* **kill** PID: sends the SIGTERM signal to the process identified by PID
* **fg**: causes a job that was stopped or in the background to return to the foreground
* **bg:** causes a job that was stopped to go to the background
* **jobs:** lists the jobs currently running or stopped
* **top:** shows the processes currently using the most CPU time (press "q" to quit)