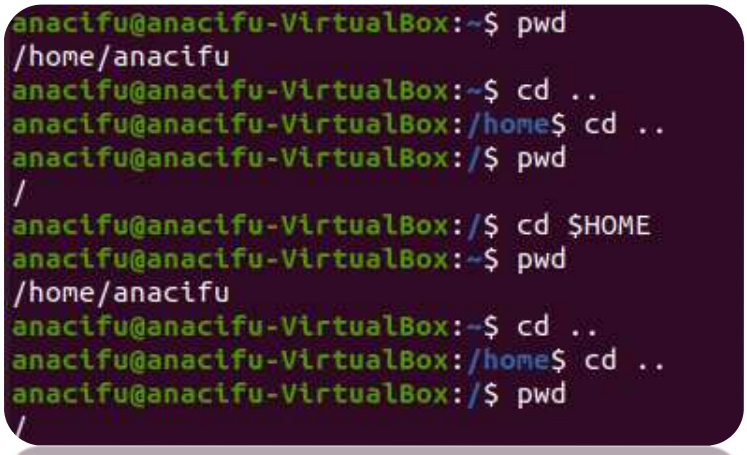
Managing directories

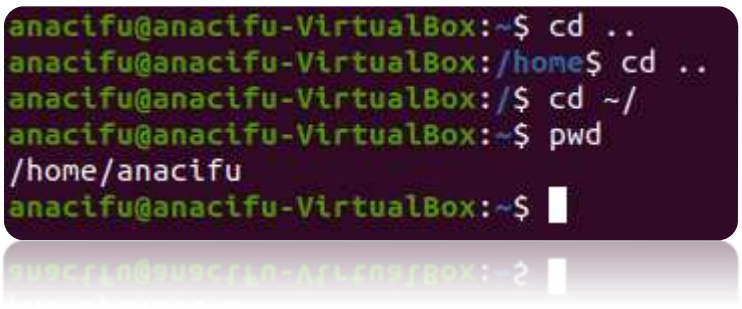
Ana Cifuentes Romero

DW1E – 18/11/2021

1. Change the current directory to /home/*<user>*.
   There are several ways to access our home directory directly:

   $ cd $HOME
   $ cd ~
   $ cd – -
   $ cd
   $ cd ~/
   $ pwd (shows the current directory)





The command **cd (change directory)** allows us to change the directory in which we are to a different one. It is a command integrated into the system, that is, there is no need to install any additional package, since it is a command belonging to the Shell of the operating system. To change directory, **cd** allows us to use absolute or relative paths.

The absolute path to a directory indicates all the directories to go through starting from the root of the file system **(/)**.

The path relative to a directory indicates the path to the directory to which we want to change but based on the directory from which the command is executed, that is, from the current or working directory.

The parent directory is the one that contains the current or working directory. To build relative paths you need to know that **..** indicates the parent directory.

If we run **cd ..** What we're doing is switching to the parent directory of the current directory, a directory just before in the directory tree.

. : Refers to the directory where we are. When we run a program and write. /program.sh, that point in front of the executable indicates that it is in the working directory.

.. : Refers to the parent directory. Using **..** we can tell the cd command to level back.

/home: It is the directory of standard users, and therefore, intended to store all user files, such as documents, photos, videos, music, templates, etc. It also includes temporary files of applications run in user mode, that serve to save program settings, etc.

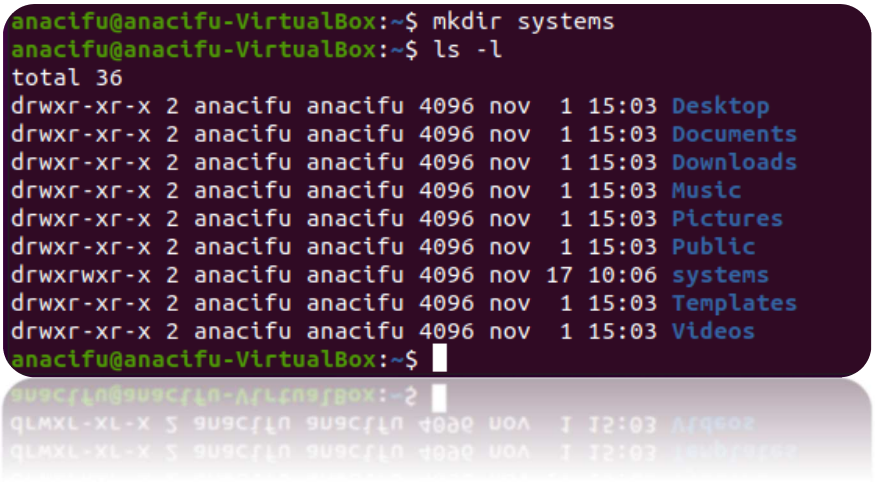
Inside /home are the personal directories of all users, named according to the username used.

$HOME: Stores the absolute path to your home directory. This turns out to very handy when you need the path of your home directory in shell scripts.

2. Create a directory called systems.

Suppose we are in the user's home directory (:~$) and we want to create the **systems** directory, the command would be as follows:

$ mkdir systems

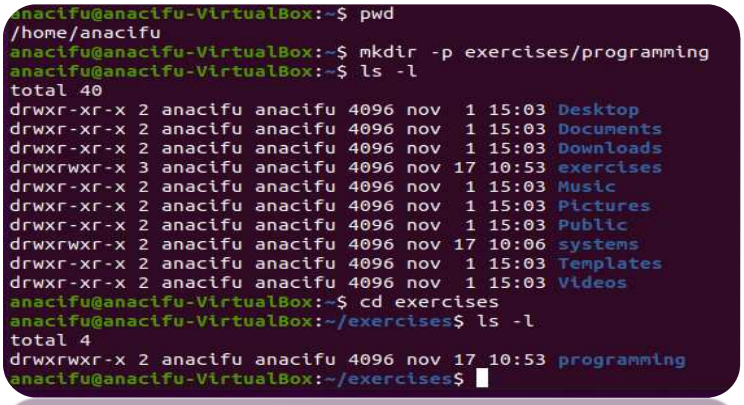


The **mkdir** command is the command used to create directories or folders on Linux

3. Create a directory called **exercises**. Go to this directory and create another new directory called **programming**.

   From /home directory: creates the exercises directory, then inside this directory creates the programming subdirectory.
   $ mkdir exercises; cd exercises; mkdir programming

   Suppose now that we want to create the programming subdirectory that hangs from exercises, the command would be the following.
   $ mkdir -p exercises/programming



-p, -parents: If the directory indicated in the path does not exist, create it. It is very useful if you want to create all the directories in the path at once. If they exist, it does not generate error.

4. Go back to the user's home directory. Delete the directories **"exercises"** and **"programming"** using just one command. Create the directories again with one command.
   $ rm -rf exercises
   $ mkdir -p exercises/programming

rm –rf: Remove the directory "exercises", and any files and directories it contains. If a file or directory that **rm** tries to delete is write-protected, you will not be prompted to make sure that you really want to delete it, just do id.

This really does remove one or more files just by specifying their name. At the same time, we can delete a complete directory if we write the -r (**r:** Recursion means that the command is used not only in the directory to which it applies, but also in its subdirectories.) option below. It is one of the most dangerous commands, which you should be very careful to use especially as root. **rm -rf exercises** to delete files or directories forcefully, you can use this option to force a deletion operation without rm asking for confirmation.

```
acifu@anacifu-VirtualBox:~$ pwd
/home/anacifu
anacifu@anacifu-VirtualBox:~$ ls -l
total 40
drwxr-xr-x 2 anacifu anacifu 4096 nov  1 15:03 Desktop
drwxr-xr-x 2 anacifu anacifu 4096 nov  1 15:03 Documents
drwxr-xr-x 2 anacifu anacifu 4096 nov  1 15:03 Downloads
drwxrwxr-x 3 anacifu anacifu 4096 nov 17 10:53 exercises
drwxr-xr-x 2 anacifu anacifu 4096 nov  1 15:03 Music
drwxr-xr-x 2 anacifu anacifu 4096 nov  1 15:03 Pictures
drwxr-xr-x 2 anacifu anacifu 4096 nov  1 15:03 Public
drwxrwxr-x 2 anacifu anacifu 4096 nov 17 10:06 systems
drwxr-xr-x 2 anacifu anacifu 4096 nov  1 15:03 Templates
drwxr-xr-x 2 anacifu anacifu 4096 nov  1 15:03 Videos
anacifu@anacifu-VirtualBox:~$ rm -rf exercises
anacifu@anacifu-VirtualBox:~$ ls -l
total 36
drwxr-xr-x 2 anacifu anacifu 4096 nov  1 15:03 Desktop
drwxr-xr-x 2 anacifu anacifu 4096 nov  1 15:03 Documents
drwxr-xr-x 2 anacifu anacifu 4096 nov  1 15:03 Downloads
drwxr-xr-x 2 anacifu anacifu 4096 nov  1 15:03 Music
drwxr-xr-x 2 anacifu anacifu 4096 nov  1 15:03 Pictures
drwxr-xr-x 2 anacifu anacifu 4096 nov  1 15:03 Public
drwxrwxr-x 2 anacifu anacifu 4096 nov 17 10:06 systems
drwxr-xr-x 2 anacifu anacifu 4096 nov  1 15:03 Templates
drwxr-xr-x 2 anacifu anacifu 4096 nov  1 15:03 Videos
anacifu@anacifu-VirtualBox:~$ mkdir -p exercises/programming
anacifu@anacifu-VirtualBox:~$ ls -l
total 40
drwxr-xr-x 2 anacifu anacifu 4096 nov  1 15:03 Desktop
drwxr-xr-x 2 anacifu anacifu 4096 nov  1 15:03 Documents
drwxr-xr-x 2 anacifu anacifu 4096 nov  1 15:03 Downloads
drwxrwxr-x 3 anacifu anacifu 4096 nov 18 11:19 exercises
drwxr-xr-x 2 anacifu anacifu 4096 nov  1 15:03 Music
drwxr-xr-x 2 anacifu anacifu 4096 nov  1 15:03 Pictures
drwxr-xr-x 2 anacifu anacifu 4096 nov  1 15:03 Public
drwxrwxr-x 2 anacifu anacifu 4096 nov 17 10:06 systems
drwxr-xr-x 2 anacifu anacifu 4096 nov  1 15:03 Templates
drwxr-xr-x 2 anacifu anacifu 4096 nov  1 15:03 Videos
anacifu@anacifu-VirtualBox:~$ cd exercises
anacifu@anacifu-VirtualBox:~/exercises$ ls -l
total 4
drwxrwxr-x 2 anacifu anacifu 4096 nov 18 11:19 programming
anacifu@anacifu-VirtualBox:~/exercises$
```

5. Print the content of the root directory.

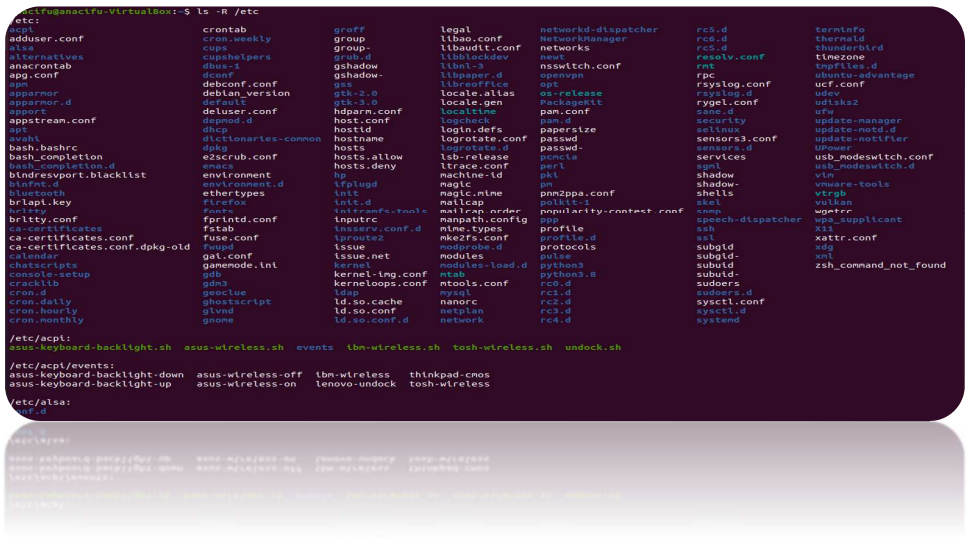
   $ls /

The **ls** command is one of the basic commands, and its main function is to list files and directories alphabetically by default. List the contents of your home directory by adding a tilde after the **ls** command.

6. Print all the files and directories that begin with d in the /usr/bin directory.
   $ ls /usr/bin/d*

***** is a wildcard when it comes to applying batch actions to multiple files and invokes all files and directories starting with **d** in this case.

7. List all the files and directories in /etc, including subdirectories.
   $ ls -R /etc

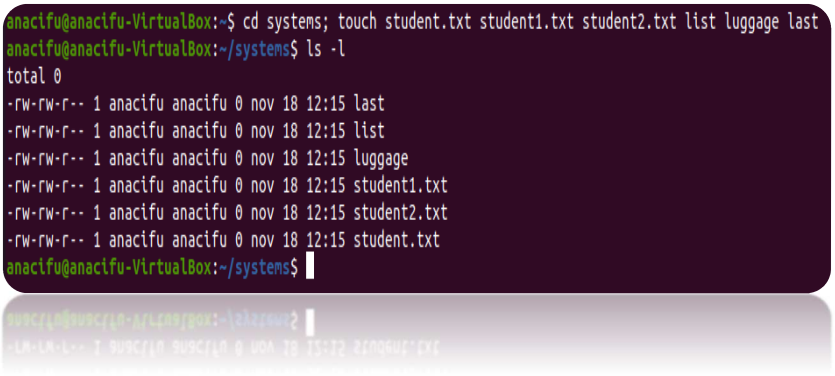


-R means show subdirectories recursively.

Recursion means that the command is used not only in the directory to which it applies, but also in its subdirectories.

In the case of the **ls** command, the recursion is applied with the **-R parameter**, since the -r (lowercase) parameter already has the use of listing the files and directories in reverse order.

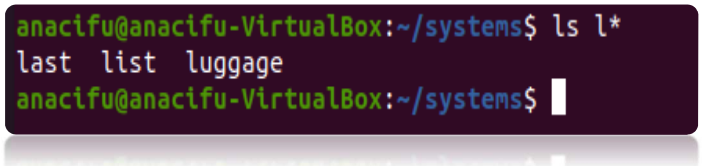
8. Create the empty files called student.txt, student1.txt, student2.txt, list, luggage and last in the directory named “system”.
   $ cd systems; touch student.txt student1.txt student2.txt list luggage last



   cd systems;: to switch to the systems directory, the semicolon allows you to execute multiple commands in succession, regardless of whether each previous command succeeds.

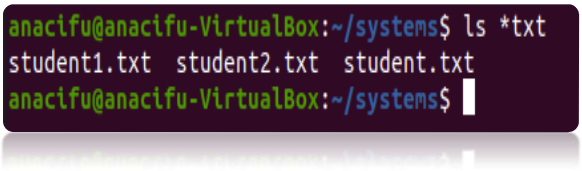
   The **touch** command is mainly used to create empty files and change timestamps of files or folders (to manipulate the access time and modification of files by using various options). The touch command without any option creates a new file. If the file exists, the command will update the access and modification time to the current time without changing its content.
   It is also possible to create multiple files using a single touch command. To do this, specify the file names with spaces between them.

9. Print all the files that begin with “l” in the directory “systems”.
   $ ls l*



10. Print all the files that end with “txt” in the directory “systems”.

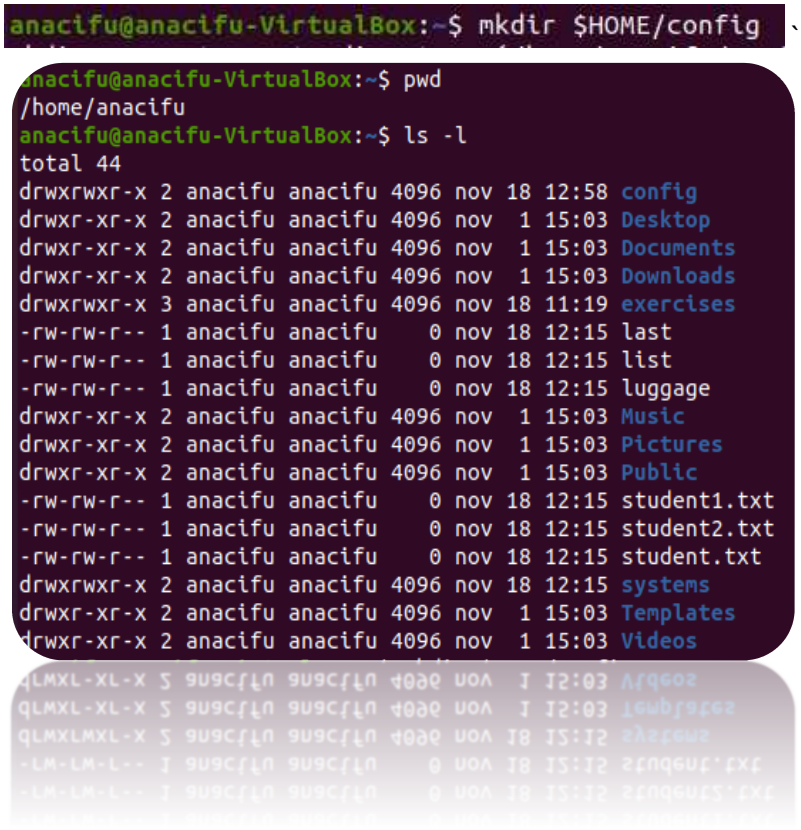
    $ ls *txt



***txt:** displays a list of txt archives.

11. Create a new directory into $HOME called “config”.

    $ mkdir $HOME/config

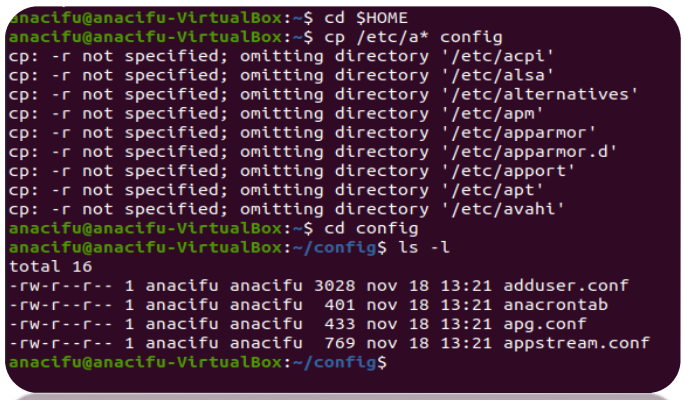


12. Copy all the files in /etc which begin with “a” into “config”.

    Frome $HOME directory:

    $ cd $HOME

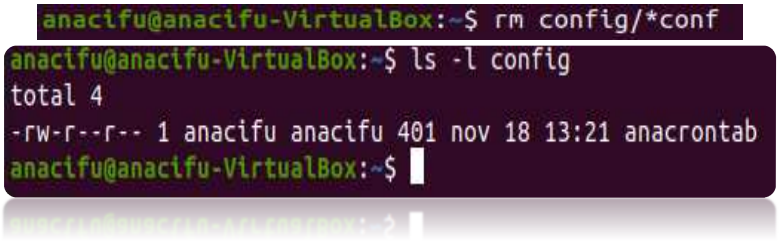
    $ cp /etc/a* config



cp stands for copy. This command is used to copy files or group of files or directory. It creates an exact image of a file on a disk with different file name. **cp** command require at least two filenames in its arguments. cp [options] source destination.

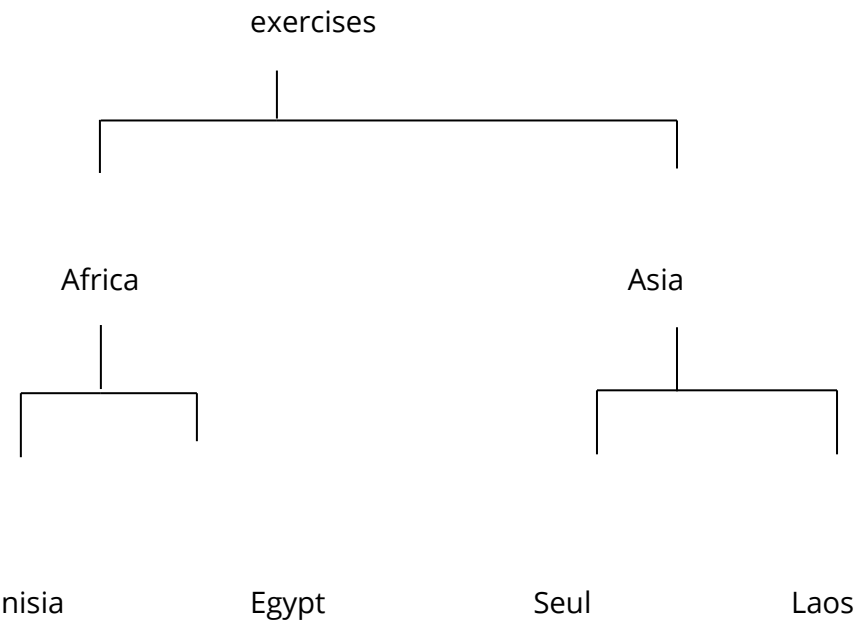
cp /etc/a* config: To copy files starting with a* from the etc directory to the config directory. If you wanted a directory from one place to another, you would have to use the -r or -R option.

13. Delete all the files in “config” which end with “conf”.
    From $HOME
    $ rm config/*conf

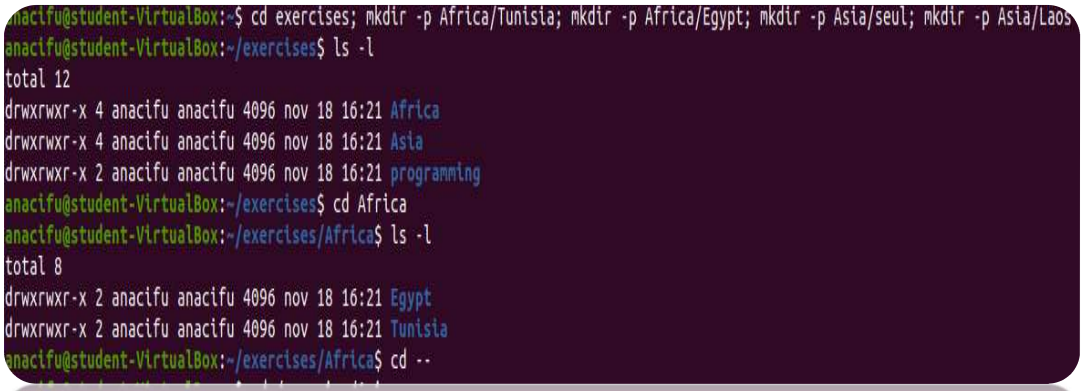


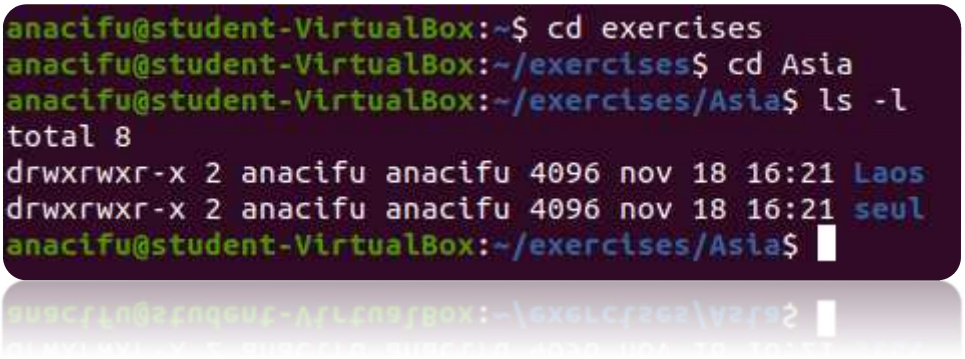
B. COPY, RENAME AND EDIT FILES

Create the following directory structure in /home/<user>/exercises.



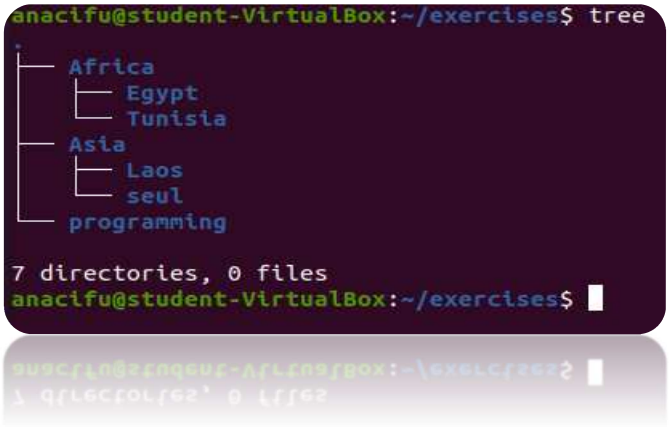
$ cd exercises; mkdir -p Africa/Tunisia; mkdir -p Africa/Egypt; mkdir -p Asia/Seul; mkdir -p Asia/Laos





-p, -parents: If the directory indicated in the path does not exist, create it. It is very useful if you want to create all the directories in the path at once. If they exist, it does not generate error. Then directory/subdirectory.

cd exercises;: to switch to the systems directory, the semicolon allows you to execute multiple commands in succession, regardless of whether each previous command succeeds.



14. Create the following files using **nano** (or another editor): **song.doc** (write a few lines of a song) in the directory **Egypt**; **tv.doc** (write the name of your favourite TV programs) in the directory **Seul** and **marks** (write the marks you expect to get this year) in the directory **Tunisia**.
    This is created from $HOME:

    $ nano Africa/Egypt/song.doc
    $ nano Asia/Seul/tv.doc
    $ nano Africa/Tunisia/marks

The **tree** command also lists the directories of external devices.

Displaying the directories in the form of a tree in Linux is quite simple, thanks to the tree utility, which, by default, is not installed in most Linux distributions but which is in the official repositories.

It is a command widely used by Linux users, which allows us to display graphically and in a structured way the hierarchy of the directories of our operating system.

The way to create files is simply by invoking a text editor, this can be the **nano, vi, gedit** or any other that we want. We'll use the **nano** text editor.

To create the file all we have to do is write the **text editor** used and the relative or absolute path of the file (we include the file). When we do the terminal will change to the **text editor** (because we are using the nano), write what you want to write and save it, then the file will already be created in the path we have chosen. Depending on the **text editor** used, the reaction of the call will be different.

CHANGE THE CURRENT DIRECTORY TO AFRICA AND DO THE FOLLOWING EXERCISES USING RELATIVE PATHS

15. Copy the file “song.doc” into “Laos”.
    $ cp Egypt/song.doc ../Asia/Laos

cd Africa;: to switch to the systems directory, the semicolon allows you to execute multiple commands in succession, regardless of whether each previous command succeeds.

cp stands for copy. This command is used to copy files or group of files or directory. It creates an exact image of a file on a disk with different file name. **cp** command require at least two filenames in its arguments. cp [options] source destination.

The path relative to a directory indicates the path to the directory to which we want to change but based on the directory from which the command is executed, that is, from the current or working directory.
The parent directory is the one that contains the current or working directory. To build relative paths you need to know that **..** indicates the parent directory.
If we run **cd ..** What we’re doing is switching to the parent directory of the current directory, a directory just before in the directory tree.
. : Refers to the directory where we are. When we run a program and write. /program.sh, that point in front of the executable indicates that it is in the working directory.
.. : Refers to the parent directory. Using **..** we can tell the cd command to level back.

The **tree** command also lists the directories of external devices.

16. Copy the file “tv.doc” into “Egypt” with the name “radio.txt”

$ cp ../Asia/Seul/tv.doc Egypt/radio.txt

Copy from the relative path directory Asia, subdirectory Seul the file tv.doc to the current directory Africa, subdirectory Egypt with the name radio.txt. Then you can show the tree and edit the text in the command line with **“tail”** command that allows to see part of the information of a file and the argument **"n"** in which we can immediately indicate the number of rows that will be displayed from the end.

17. Rename the file “marks” to “exam.wri” (it is located in the directory named Tunisia)

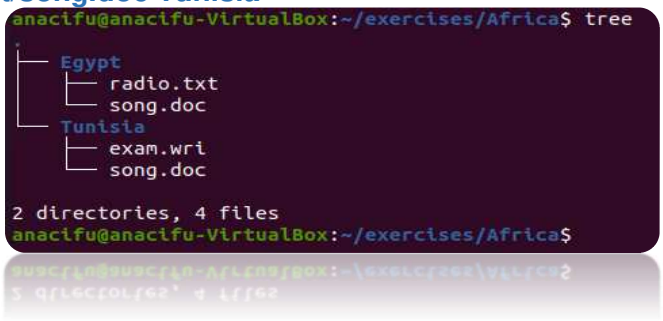
$ mv Tunisia/marks Tunisia/exam.wri

The “**mv**” command is a command line utility that moves files or directories from one place to another. It supports moving single files, multiple files and directories. It can prompt before overwriting and has an option to only move files that are new than the destination.

To move a file using the mv command pass the name of the file and then the new name for the file. In the following example the file **marks** is renamed to **exam.wri** into the same directory.

18. Copy the file “song.doc” (located in “Egypt”) inside “Tunisia”

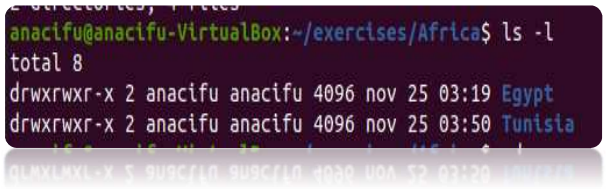
$ cp Egypt/song.doc Tunisia



Using the **“cp”** command you can copy from subdirectory **Egypt** the file **song.doc** to another directory.

19. Print the content of the current directory using long listing format.

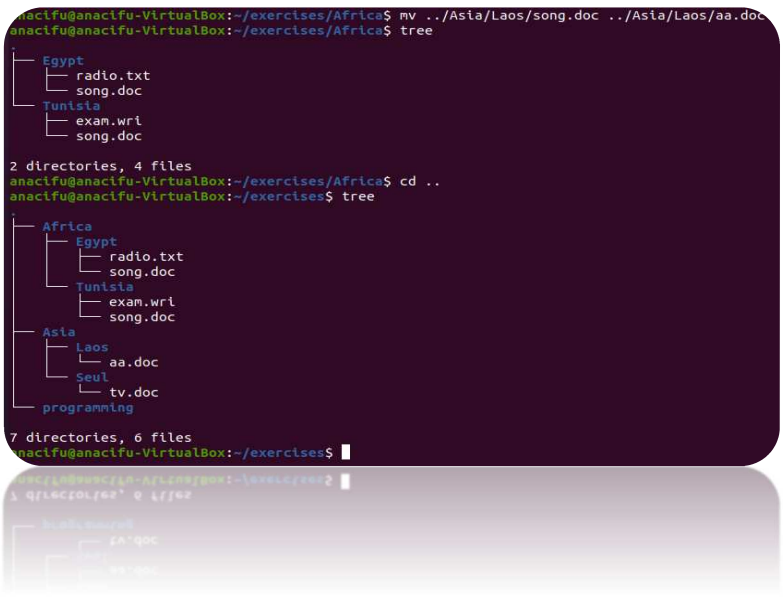
$ ls -l



The **ls** command allows viewing the contents of the directory in which we are located and the **-l** (long) option of the ls command provides more information than the name of the files such as the attributes of the permissions, the owner of the file, group of users, size in bytes, etc...

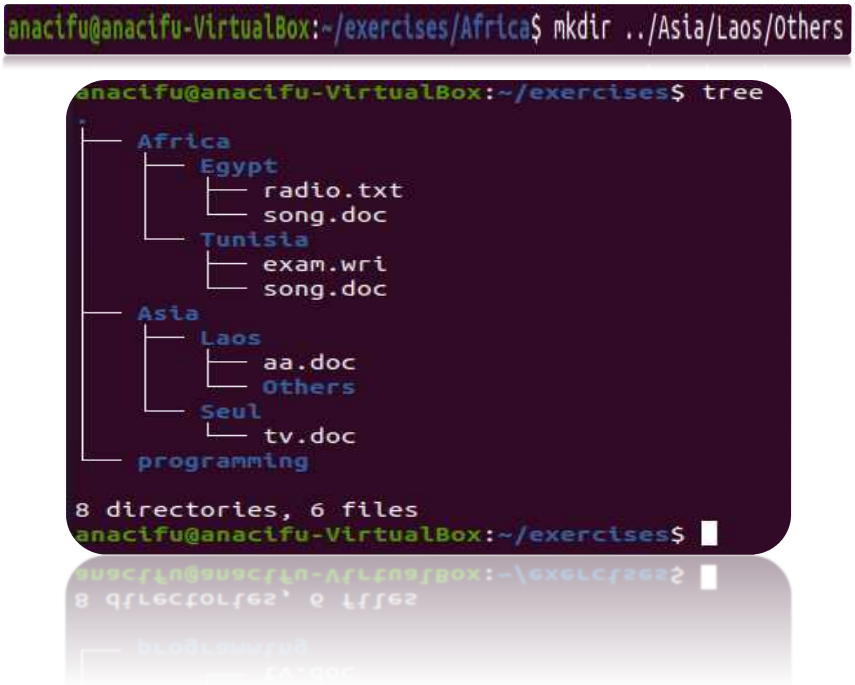
20. Rename the file “song.doc” (located in “Laos”) to “aa.doc”.

$ mv ../Asia/Laos/song.doc ../Asia/Laos/aa.doc



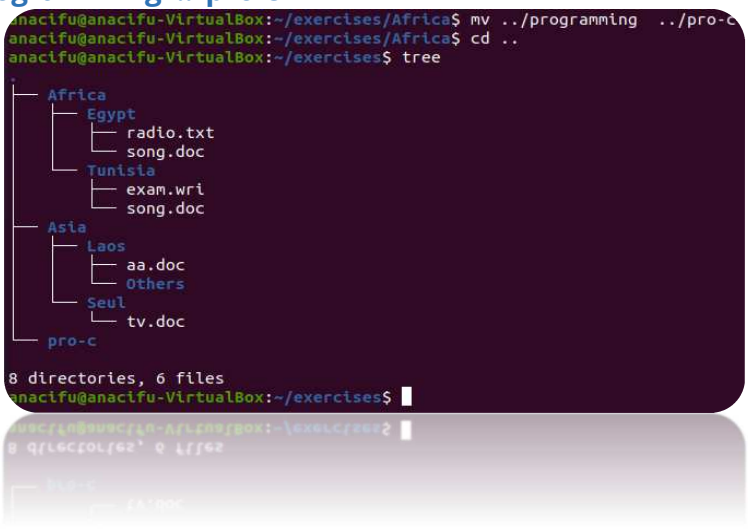
With the **mv** command you can rename into the same directory with another name taking into account that Laos is a subdirectory inside another directory.

21. Create the directory “Others” in “Laos”.
    $ mkdir ../Asia/Laos/Others



In this case you use the **mkdir** command to create a directory indicating that you want to create a directory called Others which is in another directory which in turn is in another directory.

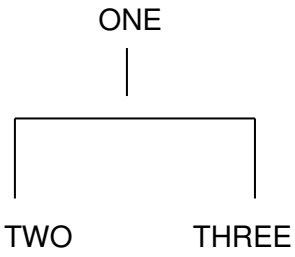
22. Rename the directory named “Programming” (which you created in the first exercises) to pro-c.
    $ mv ../programming ../pro-c



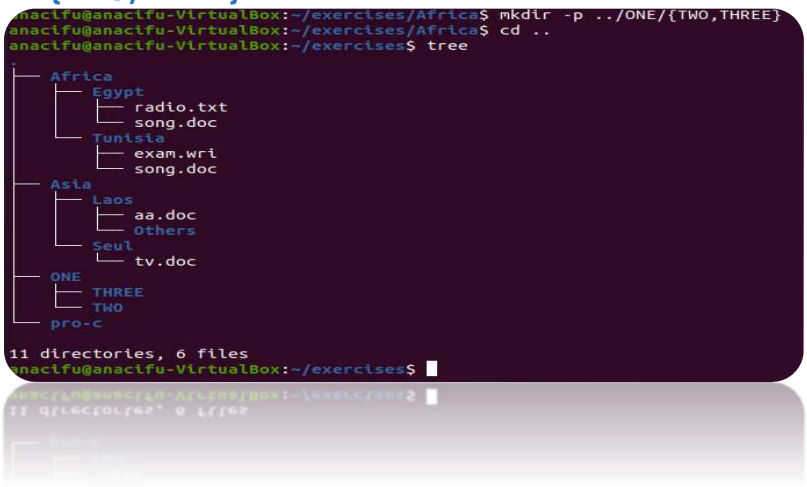
The **“mv”** command is a command line utility that moves files or directories from one place to another. It supports moving single files, multiple files and directories. It can prompt before overwriting and has an option to only move files that are new than the destination.

With the mv command you can rename a directory taking into account that the directory **“programming”** belongs to another directory inside **”exercises”** directory.

23. Create the following directory tree in “exercises” (using as few commands as possible).



$ mkdir -p ../ONE/{TWO,THREE}

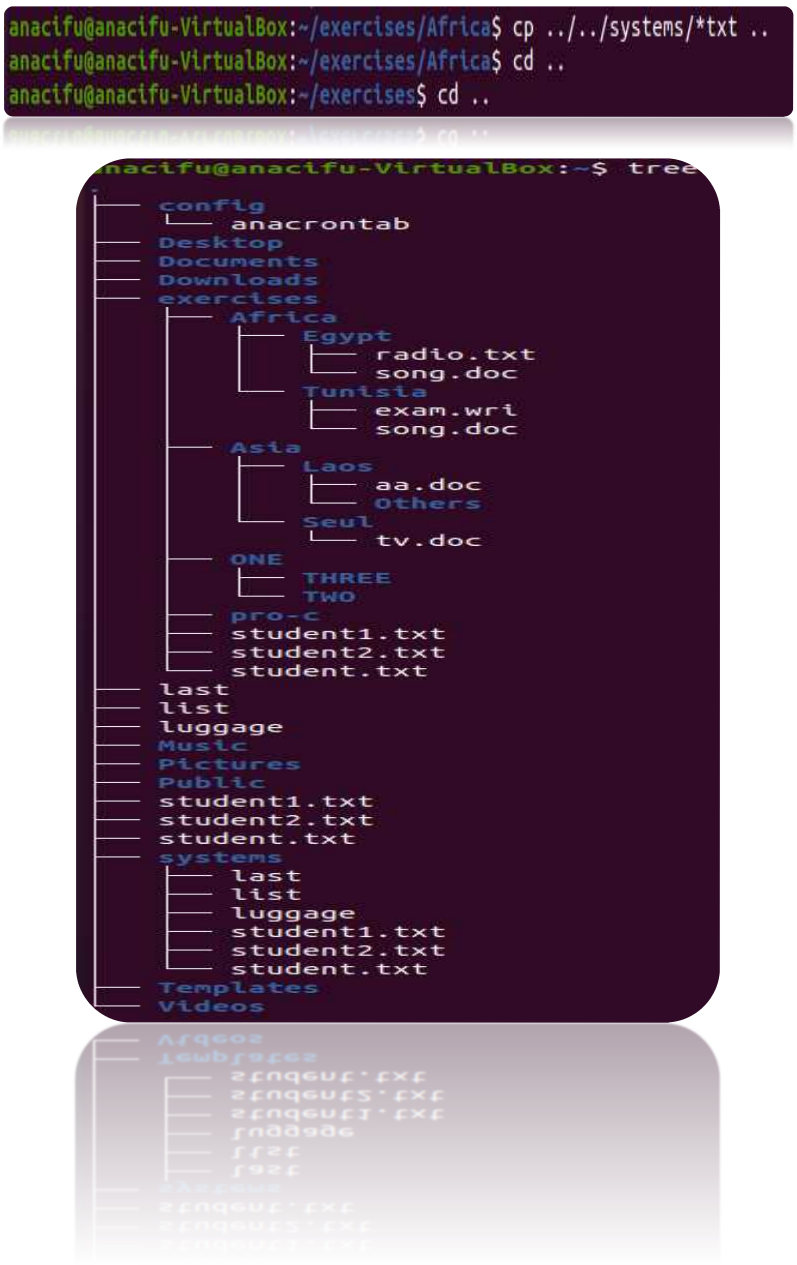


The **mkdir** command is used to create directories or folders on Linux**.**

-p, -parents: If the directory indicated in the path does not exist, create it. It is very useful if you want to create all the directories in the path at once. If they exist, it does not generate error. Then directory/ and two subdirectories in “exercises” directory. **This is a new branch of the tree**.

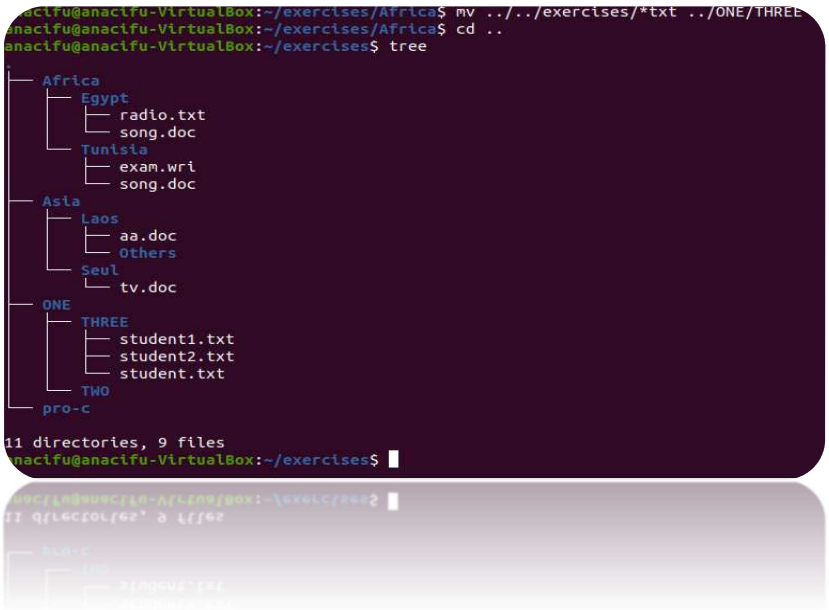
To create a new directory with multiple subdirectories faster the **{}** are used.

24. Copy the files with txt extension from de directory named “systems” to “exercises”.
cp ../../systems/*txt ..



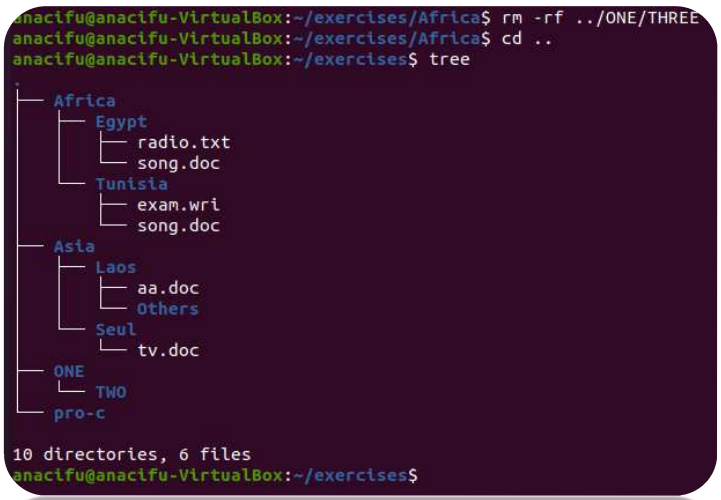
This means that you are coping the **txt** extension files from the **"systems"** directory which are inside /home/anacifu/systems to **"exercises"** directory which is another branch of the tree.

25. Move the files you copied in the exercise above to the directory THREE.
    $ mv ../../exercises/*txt ../ONE/THREE



This means that you are coping the **txt** extension files from the **"systems"** directory which are inside /home/anacifu/systems to **"THREE"** directory which is another branch of the tree.

26. Delete the directory THREE including all the files and, if so, subdirectories.
    $ rm -rf ../ONE/THREE



rm –rf: Remove the directory **"THREE"**, and any files and directories it contains. If a file or directory that **rm** tries to delete is write-protected, you will not be prompted to make sure that you really want to delete it, just do id. From the current directory **"Africa"** to another subdirectory of a directory belonging to another branch of the tree.