***LINUX EXERCISES - FILE MANIPULATION***

**PART 1:**

Create a file called “awards” in your home directory, containing

JESSE#EISENBERG#THE SOCIAL NETWORK#ACTOR

COLIN#FIRTH#THE KING’S SPEECH#ACTOR

HALLE#BERRY#FRANKIE AND ALICE#ACTRESS

NATALIE#PORTMAN#BLACK SWAN#ACTRESS

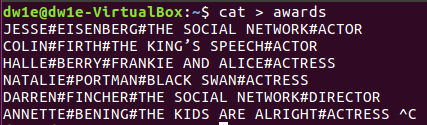
DARREN#FINCHER#THE SOCIAL NETWORK#DIRECTOR

ANNETTE#BENING#THE KIDS ARE ALRIGHT#ACTRESS

**<pwd>**

**<cat > awards>** and copy the text.





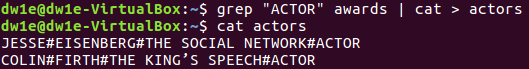
1. Add a new line, containing CHRISTOPHER#NOLAN#INCEPTION#DIRECTOR, at the end of the file. You should complete this exercise using commands instead of file editors.

**<cat >> awards>** and concatenate the rest of the text.



1. Create a file called “actors” in which you only include ACTORS.

**<grep “ACTOR” awards | cat > actors>**



EASIER WAY: We can skip using the pipe if we use:

**<grep #ACTOR$ awards > actors>**

\*\*and we should take more into account the regular expressions (in this case **#** and **$**).

1. Display the files which begin with “a” in your home directory.

**<ls a\*>** (located in our $HOME directory)



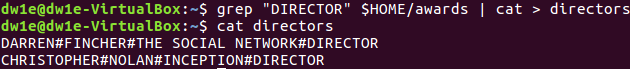
Also seen:

**<ls -d $HOME/a\*>**

**<ls $HOME | grep “^a”>**

1. Create a file called “directors” in which you only include DIRECTORS.

**<grep “DIRECTOR” $HOME/awards | cat > directors>**



\*\*we need to specify the file (“awards”) because we are first filtering from a file and then creating another one, but as we are currently in $HOME, typing “$HOME” is not needed in this case

EASIER WAY: again, we can avoid using the pipe with:

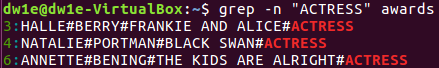
**<grep “DIRECTOR$” awards > directors>**

**→ as I repeated this in the following exercises, it won’t be covered anymore**

and we should take into account regular expressions (**$** in this case).

1. Display the lines of “awards” containing actresses. Show the line number.

**<grep -n “ACTRESS” awards>**



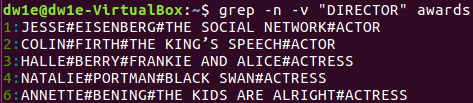
\*\* -n to show the number

**\*\***use regular expressions → **<grep -n “#ACTRESS$” awards>**

\*\*SEEN IN CLASS → **<awk -F “#” ‘{print$1}’ awards>** to print the different fields (in this case the first ($1)) for each of the lines separated with “#”. **<awk>** also supports conditional structures.

1. Display the lines in “awards” which are not directors. Show the line number.

**<grep -n -v “DIRECTOR” awards>**



\*\* -n to show the number

\*\* -v to show the lines that do NOT have “DIRECTOR”

**\*\***use regular expressions → **<grep -nv “#DIRECTOR$” awards>**

**PART 2:**

Create the file called “people.txt”

Ana;Perrrrrrrales;1000

Pedro;Soria;1600

Jacinto;Manzano;500

Claudia;Pastor;600

Xavier;Granados;1000

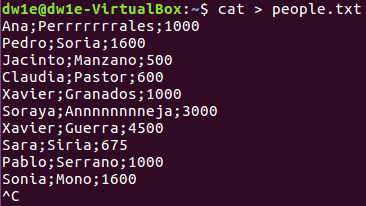
Soraya;Annnnnnnneja;3000

Xavier;Guerra;4500

Sara;Siria;675

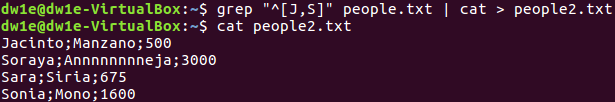
Pablo;Serrano;1000

Sonia;Mono;1600



1. Create a file called “people2.txt”, containing those people whose name begins with J or S.

**<grep “^[J,S] people.txt | cat > people2.txt”>**



Also seen:

**<grep -i “^j\|^s”> →** (\) end of the line

My way can also work without “,” between brackets:

**<grep “^[JS]” people.txt > people2.txt>**

Sergio’s way:

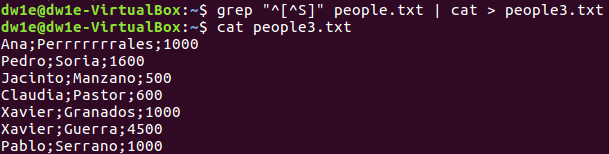
**<grep “^J”> people.txt > people2.txt**

**<grep “^S”> people.txt > people2.txt**

**→** These 2 commands together establish an order (first J and then S).

1. Create a file called “people3.txt”, containing those people whose names do not begin with S.

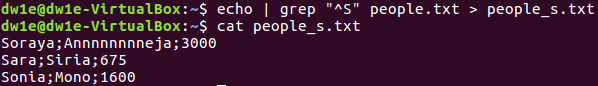
**<grep “^[^S]” people.txt | cat > people3.txt>**

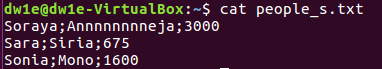


Also seen: **<grep -v “^S” people.txt > people3.txt>**

1. Display people whose name begin with S and redirect to file to “people\_s.txt”

**<echo | grep “^S” people.txt > people\_s.txt>**

CORRECTION: using **<echo>** here is not correct (even though it works). The correct way: **<grep “^S” people.txt > people\_s.txt> :** 

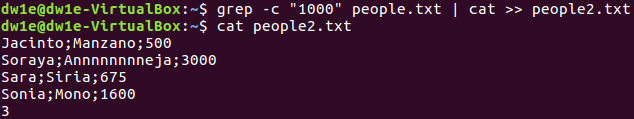


1. Display the number of people whose name begins with A (case insensitive).



1. Display how many people earn 1000 and concatenate the result in people2.txt

**<grep -c “1000” people.txt | cat >> people2.txt>**



\*\*use regular expressions → **<grep -c “1000$” people.txt >> people2.txt>**

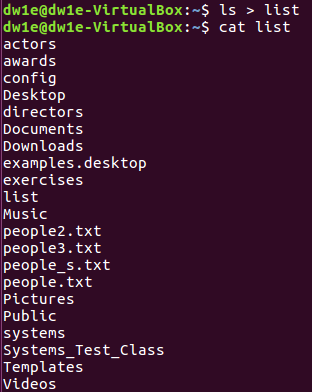
**PART 3:**

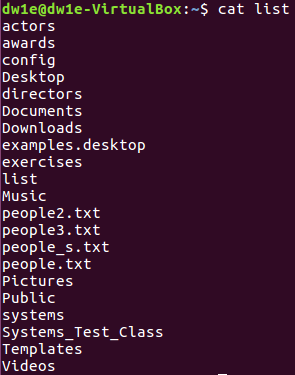
1. Create a file called list, including the contents from the current directory.

**<pwd>**

**<ls > list>**



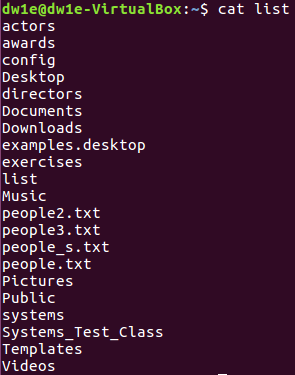




EASIER WAY: **<ls -R > list>**

1. Open the file to check if the content is right using cat, more and less and observe the differences

**<cat list>**



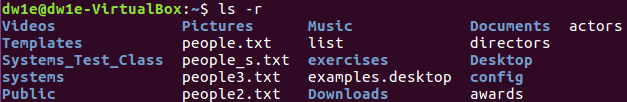
Also seen:

**<more list> →** enter to go forward (you cannot go back) → you can finish pressing Q

**<less list>** → you can go back

1. Print the contents of the current directory in reverse alphabetical order.

**<ls -r>**



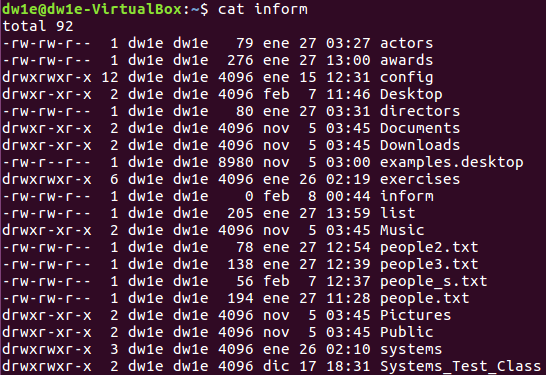
**<ls | sort> →** alphabetical

**<ls | sort -r> →** reverse alphabetical

1. Create a file called inform, containing the long format list of the files and directories in your home directory

**<ls -l $HOME > inform>**

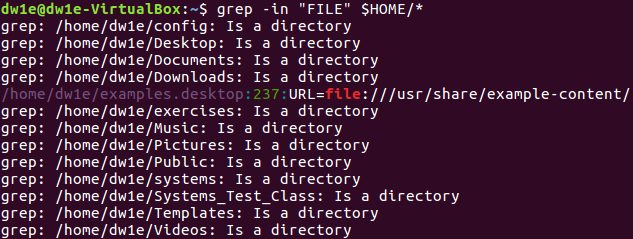




(and more below...)

1. Find the word FILE in each file of your home directory, ignoring case and showing the line number (create files containing this word if you want any match).

**<grep -in “FILE” $HOME/\*>**



Also seen:

**\*\*** with ***-w*** it only prints the lines which have that complete word

**<echo “FILE” > text1.txt>**

**<grep -in “FILE” $HOME>\*\***

**ALFREDO PUERTA GALLEGO DW1E**