Piscine C-Day 01-xinxu-0822

9:55-10:25 include doing 42 push ups
videos

1.ECHO, CAT, MORE—display

yesterday you have covered basic shell behaviours—how to move around directories, how to create directories and files, how to rename files (mv, move) extra.
today focus is file content processing: how to display, filter and even to modify. + sth else that is very important—Redirection—which you will use a lot not only during this camp but also during your entire life as a developer
Lets start with display command lines, that will display things in the standard output. I'll explain what i mean by 'things' as I go.
1st basic command line is echo, it is a "bonjour" command that write arguments to standard output. It can take more than one argument into account. you may wonder whats the relevance of echo, we will get to that.
another useful command to display file content on screen id cat. cat take the file as an argument and display the content of the file. cat also have options, check it out. noteworthy -e, "display non-printing characters to be displayed as well as, and display a dollar sign at the end of each line"
There are two command that are very similar, less and more. they also allow you to display contents on the standard output, slightly more practical than cat. e.g.more file.txt, it doesn't display everything but you can browse its content up and down using the **arrow key**. you can also search within the file by typing in the key word. type 'Mathieu', the 1st line display the 1st occurrence of that word.
Less is similar to more, but also allow backward movement. advice u to use 'less' over 'more'. as less is more advanced than more, the feature is kept more up-to-date. Less is also integrated in the 'man' command as you can browse though the content with arrow keys.

2.HEAD, TAIL, GREP—filter

10:27-10:36
now we know how to display contents, lets start filtering that display.
we are going to filter using 3 simple command lines, two of which are very similar —head display the first part line of the file and tail display the last part of the file. e.g. head file.txt
Mind their options as well, e.g. tail -n 3 file.txt, will display the last three lines of the file.
now lets check out grep, with which you can specify which line you want to display. e.g. grep "Mathieu" file.txt, it will display all lines that content this word.
grep has lots of options, lets now explore a few. grep -v "Mathieu" file.txt will display all lines that does not include this word. grep -i "Mathieu" file.txt will give case-insensative result of Mathieu.
You will use grep a lot, check out usage online and man

3.REDIRECTIONS

lets see how we can chain these commands together, in order to do that, we need to learn some new concepts. we know how to display file and filter the display one command at a time, this video teach you how to combine multiple command lines called **chain commands**.
You probably notice that everything is displayed on the standard output, meaning being displayed on your screen.

no file behaviour if you try cat alone, there is no arrow message, is waiting for your instructions, is waiting for u to type some sort of replacement for the missing file. if you type bonjour, as soon as i type enter, it will display bonjour. basically everything it reads in the standard input will be redisplayed in the standard output, aka, on screen. type **control+D** to exit this mode. So far it isn't all that interesting, but soon you will see how to redirect those standard inputs and outputs, which will allow you to chain commands.

Output Redirections:

	[]echo “bonjour"		
	bonjour		
redirect standard output to a file	[]echo “bonjour” > output	>	use the "greater than" sign to re ile. It means, instead of display en, display into the specific file.
		output	that file need to be specified aft greater-than character
	nothing is going to happen on the screen, however		
	if you type ‘l’, you will notice that file ‘output’ have been created		
	and if we do cat on this file, e.g cat output, bonjour will be displayed on the screen		now the redirection is a success
	if you repeat echo “bonjour” > output echo “bonjour” > output echo “bonjour” > output echo “bonjour” > output for a few times and then do a car again cat output, the displayed content is:		
	bonjour		the reason why you still only hav bonjour. Why? coz everytime yo ect the same thing with “>”, it e e selected file content and repla th the output
	if u wanna add things to a file	>>	will add content to a new line

	instead of replacing it		
	[]echo “bonjour2” >> output []cat output bonjour bonjour2		

Input Redirections

<	to redirect standard input from a file, the less than character is used	
|	redirect the output of the command cat into t he input of other commands , thats where th e pipe symbol kicks in	you can add as mai
	e.g. []cat file.txt | grep “Mathieu"	grep what results fr
	e.g. []cat file.txt | grep “Mathieu” | head -n 1	if you only want to r "head -n 1" will sele

4.SORT, CUT
finished this video at 12:22
let us now look at a command called “sort”, it will allow you to sort what you given it as arguments. e.g. cat file.txt| sort, this command sort all users alphabetically
be careful, this is the sorting that we call **lexicographic**, which means it's based on **ski character sorting**, so upper case are before lower cases. (显示大写字母的排列先，然后小写字母再按alphabetic顺序排列)
To not have this behaviour, you got options to sort numbers and also options to sort in reverse, do a man sort. use sort -f, you can make it non case sensitive

Once you sort your output output: cat file.txt| sort you are able to retrieve. just the 1st name for example. In order to do that, you need the command cut, which will allow you to cut each line depending on the **delimite**r(用来分割，可以是逗号或者空格), e.g. cat file.txt| sort | cut -d, -f 1
I will be able to retrieve the field I have chosen, and if I only want the first filds, I will just add -f 1
if I add cat -e to display non-printing characters: cat file.txt| sort | cut -d, -f 1 | cat -e
A cut command is very powerful, it allows you to do many things, for example retrieve multiple fields.

string：(字符串) a sequence of character, use double quote
argument=parameter

空格的用法(Tab)：

如果是用符号symbol to connect, tab is not obligatory	1+2， 1 + 2	similar
if 1, 2 represent two arguments/ no symbol in between, tab will make a difference	1 2， 12	the tab makes a difference

command cat -e

what can cat -e do?		
without cat -e	[]cat file.txt| sort | cut -d, -f 1 xin ui yu	
with cat -e	[]cat file.txt| sort | cut -d, -f 1 | cat -e xin$ ui$ yu $	now you will notic e that there is a 空格 (space) betw een the output an d 换行键(line feed / new line, symbo l ‘In')

command cut -d

[]echo “xin, xu” xin, xu []echo “xin, xu” | cut -d, -f1	cut -d,	set “,” as the deli miter	if m “ m
	-f1	take the first field	
[]echo “xin, xu” | cut -d, -f1 xin			
if []echo “xin. xu” | cut -d. -f1 xin		now take “.” as th e delimiter and ta ke the first fild	

if []echo “xin, xu” | cut -d. -f2 xu			

that is quite practical to retrieve only info that is important to you
files need to be first separated using same delimiter
e.g. cat file.txt | sort | cut -d , -f 1-3

Use command sed to modify what we retrieved: thomas with small t to Thomas with capital T, which is a very useful command, which allow you to make modifications on data flow.
e.g. cat file.txt | sort | cut -d , -f 1 | sed “s/thomas/Thomas"

command: sed	cat file.txt | sort | cut -d , -f 1 | sed “s/thomas/Thomas"	
	“s/…/..."	mean meter
	look for man and internet for more options with sed	

you can do one out of two modifications , you call those ‘rejects’, rejects are/aren't patterns. very powerful if you know how to use it.

the last command we will see is tr

tr	cat file.txt | sort | cut -d , -f 1 | tr “é” “e"	tr takes two arguments: a characte the one we are replacing it with
	or []cat file.txt | sort | cut -d , -f 1 | tr “éx” “eX"	

5.WC, IFCONFIG, BC, FIND, ENV, EXPORT

lets check out a few new commands that will be very practical and simple, too

wc	calculate the number of lines, characters and words (hin a file
[]wc file.txt 400 405 7446 file.txt	we can see that there are 400 lines, 405 words and 7 n it

	wc can take more than one file as a parameter
[]wc *	will give lwc for all files in the current directory
[]cat file.txt | grep Thomas Thomas,ghjtk, 90 Thomas, tuyte, 18 Thomas, big,45	if i do a cat on our file and retrieve the number of Thc
[]cat file.txt | grep Thomas| wc -l 3	number of lines is practical for counting results

Command file
15:00

[]file file.txt file.txt: UTF-8 Unicode txt	give info on a file post as parameter "you can see it is magic"

Command ifconfig

[]ifconfig	you can display your IP and mac address and etc.

command bc

[]bc wertyuiosdfghjk 2 + 3 5	a calculator, can also run sin, cps etc.
[]echo “1+2” | bc 3	on top of that, it reads from the standard input

command find

find	list all files in the directory
[]find .	list all files in the current directory
[]find /usr	including files in sub-directories in usr
	find is powerful as it can al filter
[]find /usr -name “ls*"	filter by name
	filter by last modification time
	filter by size
	filter depending on it is a file or directory
read its man	also allows you to perform action on files
	display
	delete
	run other commands

TO configure Shell scripts
command env (environment)

[]env PATH=/usr……. …. USER=bocal TERM…=iTerm.app	simply a list of variables in Shell that will be automatically sen t to all your binaries and scripts
	PATH will tell the Shell where to look for binaries
what is it for	you can use it to configure your Shell scripts

command export

add a variable line	
[]export LINE=3	the LINE variable has been crea

[]env … … LINE=3	ted with a value 3
[]$LINE	if you wanna access it, just type
if we do []echo $LINE 3	if we echo it, 3 will be displayed . because $LINE can be simply r eplaced by its value

These variables should allow u to configure Shell scripts, and later on help u configure, what we call “makefiles”, change behaviour depending on environmental variables.

6.STDOUT, STDERR

16:37
welcome to the bonus part of the day
not useful for today’s exercises, but useful for your life as a developer and rest of the bootcamp
previously mentioned standard output, but actually we have

[]cat file.txt	standard output
[]cat 45678h	a file that does not exit, result in
[]cat file.txt | rev ….. …..	file being displayed in reverse
	redirect standard error channel n output
[]cat 45678h | rev cat: 45678h: No such file or directory	-
[]cat 45678h 2>&1 | rev yrotcerid ro elif hcus on :h87654 :tac	-
	identically, we can redirect anyth a file
[]cat 45678h 2> error | rev []cat error cat: 45678h: No such file or directory	asking error channel 2 to be redi

[]cat 45678h 2> error []cat 45678h 2> error []cat 45678h 2> error []cat error cat: 45678h: No such file or directory	though repeat three times, only c overwritten unless there is a >> (
[]cat 45678h 2>> error cat: 45678h: No such file or directory cat: 45678h: No such file or directory	>>, which allow you to out error
	all of these can be quite practica essages but you only want to ke part that didn't work
[]cat file.txt file.txt file.txt file.txt 45678h 2>error	thanks to redirection, u'll be able rror file

[]echo "bonjour" > /dev/null	nothing happens
[]cat file.txt file.txt file.txt file.txt 45678h 2>error cat: 45678h: No such file or directory	display only error messages
	useful for debugging , checking w hether your scripts work

---

note to do;
re-watch :"text editor" video and learn mv for renaming

man cat for options
man file
man find
man patch
man grep
man sort
check out the usage of special symbols, "", '', (), [], any universal rules?