Welcome to The Carpentries Etherpad for UCLA Unix Shell Workshop!

This pad is synchronized as you type, so that everyone viewing this page sees the same text. This allows you to collaborate seamlessly on documents.

Use of this service is restricted to members of The Carpentries community; this is not for general purpose use (for that, try https://etherpad.wikimedia.org).

Users are expected to follow our code of conduct: https://docs.carpentries.org/topic_folders/policies/code-of-conduct.html

UCLA Spring Carpentries Workshops
https://pad.carpentries.org/2021-ucla-spring

Workshop website: https://ucla-data-archive.github.io/2021-04-16-ucla/

Instructors: Tim Dennis, Lisa M.

Helpers:
Kristian Allen (Windows/OSX)
Jamie Jamison (Windows/Linux)
Geno Sanchez (MacOS)
Dave George (macOS, Linux)
Leigh Phan (MacOS)
Scott Gruber (macOS)

Zoom:

https://ucla.zoom.us/j/96245430738?pwd=anNkSVpZZ1VpTTdDVjVkNW5lY2FEZz09

Meeting ID: 962 4543 0738
Passcode: unix

Lesson: http://swcarpentry.github.io/shell-novice/

Data for today

Data to download: http://swcarpentry.github.io/shell-novice/data/data-shell.zip

Download and unzip on desktop or other folder you are able to navigate to

Sign-in

Name / pronouns / Dept / email / Favorite Sandwich & sauce
Tim Dennis / he, him / Library Data Science Center / timdennis@ucla.edu / fried oyster po-boy & crystal hot sauce
Lisa McAulay / she,her / UCLA Library / emcaulay@library.ucla.edu / Turkey + Avocado .... but my favorite sauce is citrus aoili, but I haven't had it with those ingredients
Kristian Allen / he,him / UCLA Library / kallen2@g.ucla.edu / Beef on weck + horseradish
Geno Sanchez / he:him / UCLA Library / genosanchez@library.ucla.edu / bánh mì
Dave George / he, him / UCLA Anderson / dave.george@anderson.ucla.edu / BLTAC (Bacon, Lettuce, Tomato, Avocado & Cheddar) & Good Old Mayo
Leigh Phan / she/her / UCLA Library Data Science Center / leighphan@ucla.edu / bánh mì or PB&J
Aidin Tamhidi/ he, him/ UCLA Civil and Environmental Engineering/ aidintamhidi@ucla.edu/
Vincent Chou/he,him/Alumni/vinnychou@ucla.edu/Pombazo w/whatever chili sauce they soak the bread in
Wesley Lau /he, him/ Masters Applied Statistics / wlau218@gmail.com / Fried Chicken Sandwich
Tanner Wataers / he,him/ IoES / Pastrami on rye
KaranDeep/ he, him / simple cheese sandwich with hot n sour sauce
Lucy Dunn/ she, her/Undergrad Neuroscience/ lucymdunn@g.ucla.edu/Fried Chicken Sandwich
Ana Garcia Vedrenne / she, her / EEB / garciavedrenne@ucla.edu / I am bad at having favorite anything! It changes every day. Thinking avocado toast now <-- I can relate! Variety is the spice of life :)
Nipun Batra / he,him / nipunbatra40@gmail.com / Turkey Sandwich with Ranch on the side
Zicong Feng/He,him/Master of Applied Statistics/joefeng@g.ucla.edu/ Egg & Ham Sandwich
Marietta Nieto/she,her/IT Services/meatball sandwhich
Scott Gruber / he,him/ Institute of the Environment and Sustainability / scott.gruber@g.ucla.edu / bánh mì

Schedule:

9:00 - 9:15 - Welcome & introduction of instuctors/helpers - code of conduct
9:15 - 9:20 - Class introductions (brief)
9:20-10:00 - Motivation, Files and Directories
10:00-10:10 - Break
10:10-10:30 - Working with Files and Directories
10:30-10:45 - Challenges & Solutions - 10 min in breakout rooms & 5 min solutions
10:45-11:15 - Pipes & Filters
11:15-11:30 - Pipes & Filters Challenges - 10 min in breakout & 5 for solutions
11:30-11:40 - Break
11:40-12:10 Loops
12:10-12:20 Loop challenges - Poss. homework
12:20 - 12:30 Wrap Up/Feedback

Notes

(anyone is free to add to the notes section!)
WIndows: how to access git BASH
windows key->type "git bash"
history

Terms

- **print working directory (pwd):** outputs your current location in your file system
- **change directory (cd):** The command to change locations is cd followed by a directory name to change our working directory
- **listing (ls):** list the contents of the current directory
- # : we are using the hash or pound sign to indicate a note or comment
- **flags** (aka options, or switches): start with a single dash (-) or two dashes (--), change the behaviour of a command.
- help: (**man**) or (**--help**): displays more information on how to use the command or program. MacOS will use ms, Linux/Win(git bash) will use --help.
- **clear**: clears the terminal screen. (Does not delete history).
- make directory (**mkdir**): will create an empty directory (folder)
- rename (**mv [old name] [new name]**): To rename a file or directory you have to use the move command to 'move' your old file/directory to the new name. for example:
    - **mv homework thesis**
        - This will change the name of the 'homework' directory to 'thesis'. The contents of the directory will not be affected.
- **nano**: basic text editor. It can only work with plain character data, not tables, images, or any other human-friendly media. Nano has a handy command reference on the bottom. the caret/hat is the CTRL key. Example to exit, nano says ^X, this means CTRL-X
- **touch**: **"touch filename"** will create an empty file (0 bytes) with a name of "filename"
- **pipe**: The vertical bar | between the two commands is called a pipe. It tells the shell that we want to use the output of the command on the left as the input to the command on the right.
- word count (**wc**): will count the number of words, lines, characers and byte count
- wild cards: Wildcard characters are used to define the pattern for searching or matching text on string data. Asterisk (*****) is a wildcard, which matches zero or more characters. Question Mark (**?**) is also a wildcard, but it matches exactly one character.
    - **wc -l *.txt** (this command will give you the line count of every txt file in your directory
    - **???ane.pdb** matches three characters followed by ane.pdb, giving cubane.pdb ethane.pdb octane.pdb
- **sort**: The sort utility sorts text and binary files by lines.
- **cat**: The "cat" command in Bash stands for "concatenate". cat displays the contents of its inputs.
- **head**: displays first lines of a file
- **tail**: displays the last lines of a file
- ↑ and ↓ keys will cycle through your **history**
- **cut**: the cut command is used to remove or 'cut out' certain sections of each line in the file
- loops: Loops are a programming construct which allow us to repeat a command or set of commands for each item in a list.
    - loops can be written on one line:
        - for filename in basilisk.dat unicorn.dat ; do head -3 $filename; done
    - or on multiple lines for better readability:
        - for filename in basilisk.dat unicorn.dat

- do
- head -3 $filename
- done

pwd
/users/elizabethmcaulay

cd Desktop
change directory to the Desktop folder
format:
cd directory_name_to_change_to

pwd
/users/elizabethmcaulay/Desktop

ls
data-shell
list directory command lists the folders in the Desktop. Here, the only folder is 'data-shell'

Introducing flags (minor modifications to the command)

ls -l

-l ('long' flag - shown by lowercase "L") in addition to 'ls' lists files in the directory as well as diskspace and details about each file

ls -al

cd data-shell

To clear screen:
clear

To see what commands you used in this terminal session:
history

Getting help

On Windows:
[command] --help
e.g. ls --help

On Mac:

man [command]
(e.g. man ls)
'man' is short for 'manual'

if you are stuck on any screen that doesn't allow inputs/typing
press Q for quit

Navigating within and between directories

% cd ..
# change directory to the directory above the current one (its parent directory)

% ls .
# '.' r (indicated with a dot or period) refers to the directory you are currently in

BREAK~~~~~

Moving directories and files

% mkdir tesis
% mv tesis thesis
# to rename the folder 'tesis' to 'thesis', we have to move the folder 'tesis' to the folder we originally created called 'thesis'

% ls -F

Brief high level best practices for file naming:
https://library.stanford.edu/research/data-management-services/data-best-practices/best-practices-file-naming

% nano draft.txt
# nano is a simple text editor accessible via the shell, with the command 'nano'
# the above command 'nano draft.txt' prompts to open nano and create a file called 'draft.txt'

how to install wget from command line
On osx,
1) install brew from https://brew.sh/
2) copy/paste install line from website above
2) type `brew install wget`

On Windows, it is slightly different, this blog entry has steps outlined
https://www.yinfor.com/2020/11/how-to-add-wget-command-into-git-bash.html

redirecting out put from console/screen to a file (like txt)
wordcount is wc

wc -l *.pdb > lengths.txt

puts the output to a file
read by " cat lengths.txt"

sort command is used to order things (default ascending)

sort -r # for reverse or descending order

combining pipes:
sort -n lengths.txt | head -n 1 ## bar means output of the left pipes to the right

instead of a file name, you can pipe to another command (meaning apply that command to the output of the before)

Pipe Reading Comprehension

A file called animals.txt (in the data-shell/data folder) contains the following data:

```
2012-11-05,deer
2012-11-05,rabbit
2012-11-05,raccoon
2012-11-06,rabbit
2012-11-06,deer
2012-11-06,fox
2012-11-07,rabbit
2012-11-07,bear
```

What text passes through each of the pipes and the final redirect in the pipeline below?

```
$ cat animals.txt | head -n 5 | tail -n 3 | sort -r > final.txt
```

Hint: build the pipeline up one command at a time to test your understanding

Solution:
The head command extracts the first 5 lines from animals.txt. Then, the last 3 lines are extracted from the previous 5 by using the tail command. With the sort -r command those 3 lines are sorted in reverse order and finally, the output is redirected to a file final.txt. The content of this file can be checked by executing cat final.txt. The file should contain the following lines:

2012-11-06,rabbit
2012-11-06,deer
2012-11-05,raccoon

Pipe Construction

For the file animals.txt from the previous exercise, consider the following command:

```
$ cut -d , -f 2 animals.txt
```

The cut command is used to remove or ‘cut out’ certain sections of each line in the file, and cut expects the lines to be separated into columns by a Tab character. A character used in this way is a called a delimiter. In the example above we use the -d option to specify the comma as our delimiter character. We have also used the -f option to specify that we want to extract the second field (column). This gives the following output: |

```
deer
rabbit
raccoon
rabbit
deer
fox
rabbit
bear
```

The uniq command filters out **adjacent matching lines** in a file. How could you extend this pipeline (using uniq and another command) to find out what animals the file contains (without any duplicates in their names)?

if you want to extract only the second "column" or field then you use cut -f

Solution:

```
$ cut -d , -f 2 animals.txt | sort | uniq
```

bear
deer
fox
rabbit
raccoon

on my keyboard shift+\ makes |

Loops

```
for filename in basilisk.dat unicorn.dat
do
head -3 $filename
done
```

COMMON NAME: basilisk

CLASSIFICATION: basiliscus vulgaris
UPDATED: 1745-05-02
COMMON NAME: unicorn
CLASSIFICATION: equus monoceros
UPDATED: 1738-11-24

```
for file in *.dat
do
echo $file
head -100 $file | tail -20
```

basilisk.dat
CGGTACCGAA
AAGGGTCGCG
CAAGTGTTCC
CGGGACAATA
GTTCTGCTAA
GATAAGTATG
TGCCGACTTA
CCCGACCGTC
TAGGTTATAA
GGCACAACCG
CTTCACTGTA
GAGGTGTACA
AGGATCCGTT
GCGCGGGCGG
CAGTCTATGT
TTTTCGACAC
TGGACTGCTT
CCCTTTGAGG
GTGGATTTTT
CGTAACGGGT
minotaur.dat
GGTTGGACCG
AGTGAACGAA
GCGATTGATG
CCCAGTATTG
GTTTCGGGTC
CCACCAGGAA
CAATGTTGGA
CTTTAGCACG
ACAGGAATCA
ATCGTTTCGT
AATGGTGGCA
CTTTGGGGTT
GCAGGCCCGC
CTGTGCCTTC

AACGGTCGGC
AAATAGGACA
GAGATTTCCT
GCCTCAAACA
TCGCAAGACG
ATTTATCCGG
unicorn.dat
CGGTACCGAA
AAGGGTCGCG
CAAGTGTTCC
CGGGACAATA
GTTCTGCTAA
GATAAGTATG
TGCCGACTTA
CCCGACCGTC
TAGGTTATAA
GGCACAACCG
CTTCACTGTA
GAGGTGTACA
AGGATCCGTT
GCGCGGGCGG
CAGTCTATGT
TTTTCGACAC
TGGACTGCTT
CCCTTTGAGG
GTGGATTTTT
CGTAACGGGT

loop to create copies of the .dat files:

```
for filename in *.dat
do
echo $filename
cp $filename original-$filename
done
```

basilisk.dat
minotaur.dat
unicorn.dat

```
ls
```
basilisk.dat original-basilisk.dat original-unicorn.dat
minotaur.dat original-minotaur.dat unicorn.dat

Finding and searching
http://swcarpentry.github.io/shell-novice/07-find/index.html