CSE 374 Programming Concepts and Tools

Laptops in lecture are good, actually!

Lecture 02 - Bash Shell, Commands, File Systems

Today

We'll dive into more details about Linux, the shell, and filesystems

If all this information feels overwhelming at first, don't worry!

- I will demonstrate a lot of these commands during the lecture.

Things begin to click and make sense with practice - try following along!

Review: Linux

Linux is an operating system, like Windows or MacOS

Duties of the operating system

- Manage the processes running on a computer
- Organize files/folders and control access to them

In this course, we will be using the “Bash” shell as the text based interface for Linux.

- Use `echo $SHELL` to check which shell you are using (Linux has multiple shells)

Bash Shell

Bash (Bourne-Again SHell) Language

Bash acts as a language interpreter

- Commands are analogous to *functions* with arguments
- Bash **interprets** the arguments and calls the function
- Bash also has its own variables and logic

Bash vs. Java

Bash

- Interpreted
- Esoteric variable access
- Everything is a string
- Easy access to files and programs
- Good for quick & interactive programs (like automation)

Java

- Compiled
- Highly structured and strongly typed
- Strings have library processing
- Data structures and libraries
- Good for large complex programs

As soon as a program grows to a sufficient level of complexity, you should use a *general purpose* programming language (e.g. C, Java, Python, etc).

Commands in the Shell

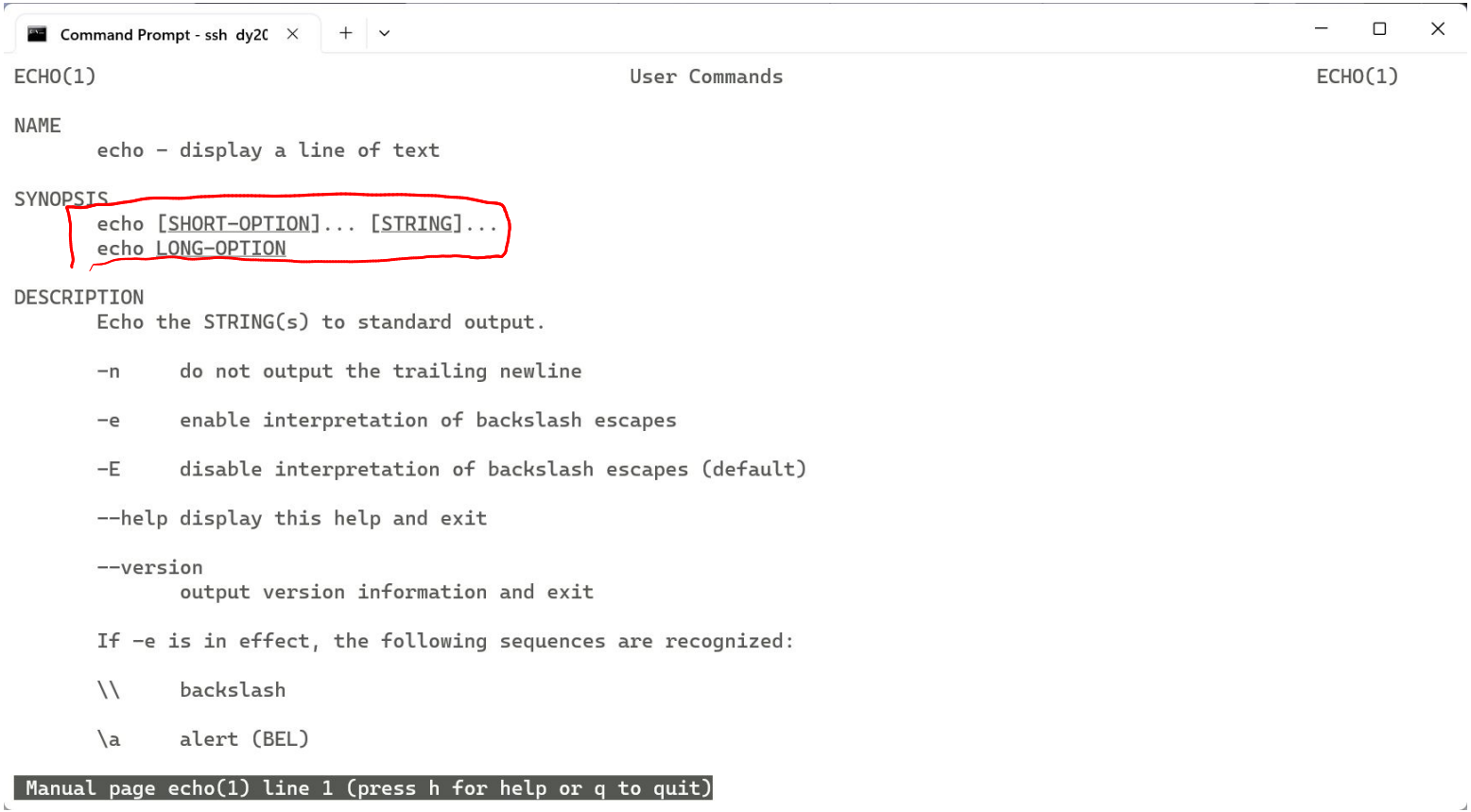
Commands are pre-existing programs

```
program_name [options] [arguments]
```

To learn about an individual command use "**man**"

- **man <command name>**
  - Short for "manual page"
  - Also available through web search
- (Usually) Can also use the **--help** option (or **-h**), e.g. **ls --help**
- Look for keyword: **man -k <search term>** e.g. **man -k list**

man echo


Command Prompt - ssh dy2(
ECHO(1) User Commands ECHO(1)
NAME
echo - display a line of text
SYNOPSIS
echo [SHORT-OPTION]... [STRING]...
echo LONG-OPTION
DESCRIPTION
Echo the STRING(s) to standard output.
-n do not output the trailing newline
-e enable interpretation of backslash escapes
-E disable interpretation of backslash escapes (default)
--help display this help and exit
--version
output version information and exit
If -e is in effect, the following sequences are recognized:
\\ backslash
\a alert (BEL)
Manual page echo(1) line 1 (press h for help or q to quit)

Poll Question

Please answer in the Poll: Lec 2 assignment on Gradescope!

`ssh` into Calgary and run **`man less`**

What does the less command do?

- Subtracts from a variable
- Removes permissions to modify a file
- Displays file contents one screenful at a time
- Shrinks a file by a specified amount

Now run **`man ls`** or **`ls --help`**

Which flag allows you to *sort files by their size*?

- `-size`
- `-sort`
- `-s`
- `-S`

Hint: you can search through the `man` page by typing **`/ <your search phrase>`** (and press enter)

Shell Interaction Basics

Copy and paste shortcuts might be slightly different (*e.g.*, Ctrl+Shift+C)

You can navigate through your executed commands by using the **up and down arrows**

- Convenient way to rerun commands or to fix small errors in previous command

You can press **Tab** to quickly finish typing an existing file/folder name

- *e.g.*, Docu+Tab ➡ Documents

Input & Output

Inputs come in two main forms: arguments and standard input (`stdin`)

Outputs are written to standard output (`stdout`) or directed to an output file

Arguments are strings separated by spaces given after the command

- EX: `wc file1 file2`
  - Arguments: "`file1`" and "`file2`"
- Arguments with spaces need to be wrapped in quotes
  - EX: `echo "hello world"`

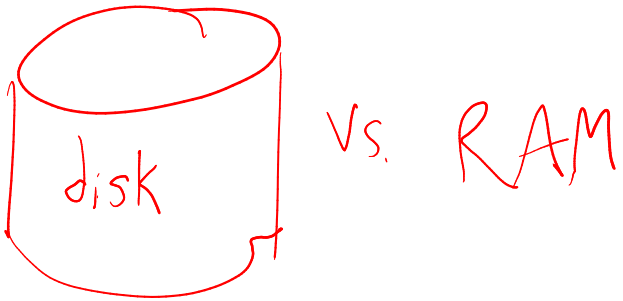
We will cover input and output streams in more detail in the next lecture!

File System

Directories

A container used to store and organize files 📂

- Stored on a hard drive AKA *disk*
- Directories are also known as *folders*



Directories have:

- **Name**: a unique string within a directory
  - e.g. “pictures”
- **Content**: files or sub-directories
  - e.g. "notes.txt”, “pictures_of_cats”
- **Location**: directory trail from drive to name (absolute vs. relative path)
  - e.g. "/Users/chloe/pictures" (absolute path)
  - e.g. “chloe/pictures” (relative path)

parent

Files

A basic unit of storage on a computer 📝

- Stored on a hard drive

Files have:

- **Name**: a unique string within a directory
  - e.g. “notes.txt”
- **Content**: data contained within the file
  - e.g. “Go to the store for milk.”
- **Location**: directory trail from drive to name (absolute vs. relative path)
  - e.g. “/Users/chloe/notes.txt” (absolute path)
  - e.g. “chloe/notes.txt” (relative path)
- **Extension:** part of the name
  - e.g. ".txt" in "notes.txt"

Unix philosophy:
"Everything is a file"

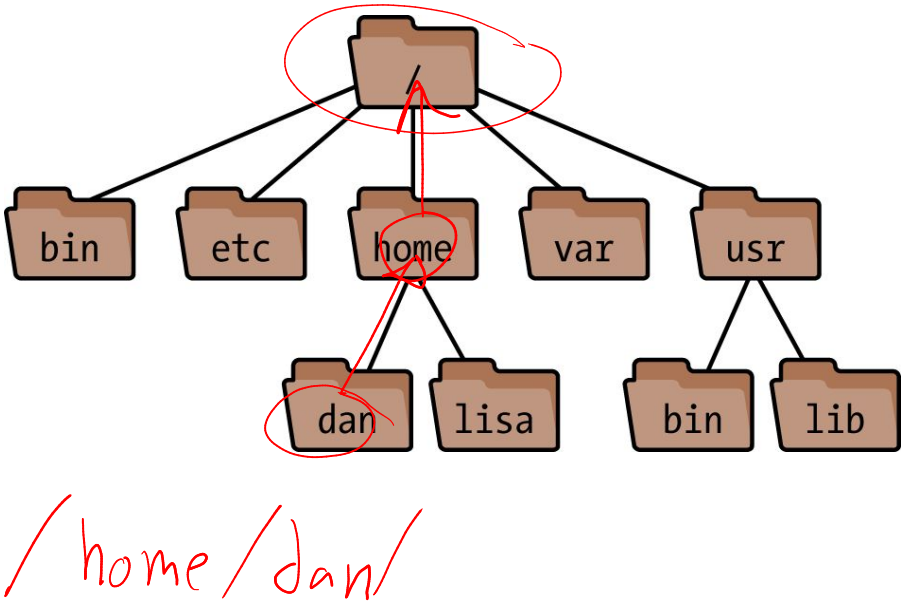
Optional!

File System

Think of the file system as a tree.

One directory may contain other directories, called subdirectories, which may themselves contain other files and subdirectories, and so on, into infinity.

The topmost directory is called the root directory and is denoted by a slash (/).



https://learning.oreilly.com/api/v2/epubs/urn:orm:book:9781491927557/files/assets/lpg3_0103.png

File Paths

We refer to files and directories using a “names and slashes” syntax called a **path**. Files can be located with a path: a list of folders to get there.

Example: **`/Users/chloe/notes.txt`**

- Look in **`/Users`**, then **`chloe`**, then you will find the file called "**`notes.txt`**"

The folders are separated by the "/" character

- The delimiter is OS-specific; Windows actually uses the “\” character.

Absolute vs Relative File Paths

Paths starting with "`/`" are called **absolute** paths

- Start searching from the root of the file system
- Absolute paths are *unambiguous*.
- e.g. "`/Users/chloe`"

Paths which do not start with "/" are called **relative** paths

- e.g. "`test/foo.txt`" or "`foo.txt`"
- Start searching from the current directory. You need to know “where you are” in the Linux filesystem.
- If we are in "`/Users/chloe/`", then "`test/foo.txt`" links to "`/Users/chloe/test/foo.txt`"

Poll Question

Please answer in the Poll: Lec 2 assignment on Gradescope!

Is the path "`foo/bar`" absolute or relative?

- Absolute
- Relative

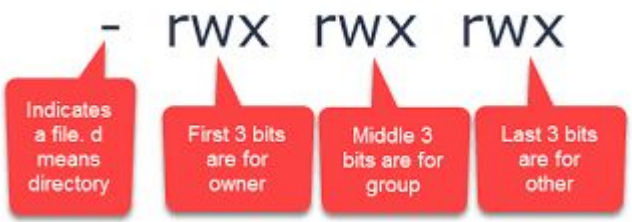
Can you tell whether `bar` is a file or a directory?

- Yes
- No

Linux File Permissions

Permission Groups

- u – User that owns the file
- g – Group that owns the file
- o – Others
- a – All users



Permission Types

- **`r`**- read: a user's ability to read the contents of the file.
- **`w`** - write: a user's capability to write or modify a file or directory.
- **`x`** - execute: a user's capability to execute a file or view the contents of a directory.

Linux File Permissions

Reading file permission output

- 1 File type: `-` = file, `d` = directory
- 2–4 Read, write, and execute permissions for the **owner**
- 5–7 Read, write, and execute permissions for the **group**
- 8–10 Read, write, and execute permissions for all **other** users
- `-rw-rw-rw-` = owner, group and all users have read & write permissions to the file

```
chmod <group>[+-=]<permission> <file>
```

- `chmod a-rw file1` : remove read and write permissions on `file1` for all users
- `chmod u+x file1` : add execute permissions on `file1` for the owner

Poll Question

Please answer in the Poll: Lec 2 assignment on Gradescope!

Given the following permissions string, answer the following questions:

```
-rwxrw-r--
```

Is this a file or directory?

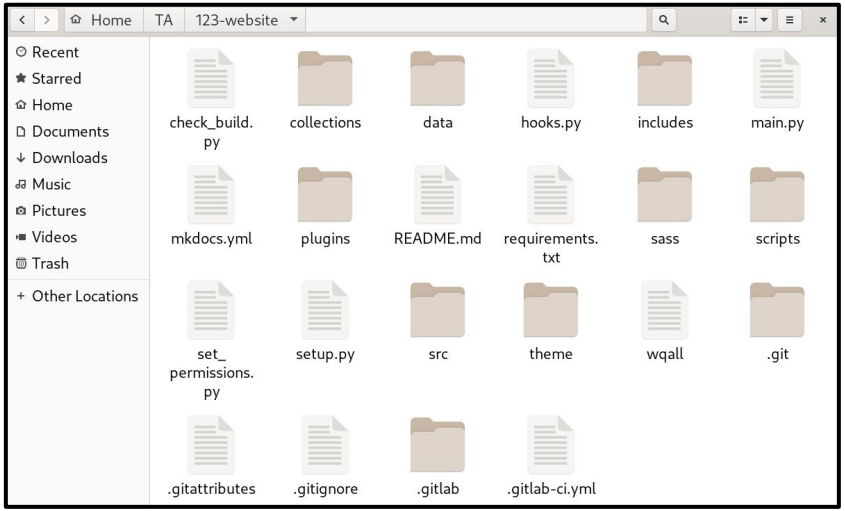
For each category—owner, group, and others—what permissions do they have?

- ❑ Read
- ❑ Write
- ❑ Execute
- ❑ None of the above

Which **`chmod`** command will give the **user and group both write and execute** permissions on **`file`** without changing any other permissions?

- `chmod a+wx file`
- `chmod +wx file`
- `chmod u+wx,g+wx file`
- `chmod ug=wx file`

```
[cf2024@attu8 123-website]$ ls -al
total 252
drwxr-xr-x 12 cf2024 vgrad_cs   24 Jun 24 17:49 .
drwxr-xr-x 17 cf2024 vgrad_cs   19 Apr  9  2023 ..
-rw-r--r--  1 cf2024 vgrad_cs  539 Apr  9  2023 check_build.py
drwxr-xr-x  4 cf2024 vgrad_cs    4 Apr  9  2023 collections
drwxr-xr-x  3 cf2024 vgrad_cs    4 Apr  9  2023 data
drwxr-xr-x  9 cf2024 vgrad_cs   17 Apr 10  2023 .git
-rw-r--r--  1 cf2024 vgrad_cs  636 Apr  9  2023 .gitattributes
-rw-r--r--  1 cf2024 vgrad_cs 9037 Apr  9  2023 .gitignore
drwxr-xr-x  3 cf2024 vgrad_cs    3 Apr  9  2023 .gitlab
-rw-r--r--  1 cf2024 vgrad_cs 2121 Apr  9  2023 .gitlab-ci.yml
-rw-r--r--  1 cf2024 vgrad_cs 3532 Apr  9  2023 hooks.py
drwxr-xr-x  2 cf2024 vgrad_cs   10 Apr  9  2023 includes
-rw-r--r--  1 cf2024 vgrad_cs 5040 Apr  9  2023 main.py
-rw-r--r--  1 cf2024 vgrad_cs 2466 Apr 10  2023 mkdocs.yml
drwxr-xr-x  2 cf2024 vgrad_cs    8 Apr  9  2023 plugins
-rw-r--r--  1 cf2024 vgrad_cs 6836 Apr  9  2023 README.md
-rw-r--r--  1 cf2024 vgrad_cs  775 Apr  9  2023 requirements.txt
drwxr-xr-x  3 cf2024 vgrad_cs   13 Apr  9  2023 sass
drwxr-xr-x  2 cf2024 vgrad_cs    4 Apr  9  2023 scripts
-rw-r--r--  1 cf2024 vgrad_cs  744 Apr  9  2023 set_permissions.py
-rw-r--r--  1 cf2024 vgrad_cs  758 Apr  9  2023 setup.py
drwxr-xr-x 17 cf2024 vgrad_cs   29 Apr  9  2023 src
drwxr-xr-x  6 cf2024 vgrad_cs   24 Apr  9  2023 theme
-rw-r--r--  1 cf2024 vgrad_cs 1453 Apr  9  2023 wqall
```



`ls -al` view vs GUI view of a directory

Special File Paths

working directory

`.` is your current directory

`..` is the directory above the current directory, parent directory

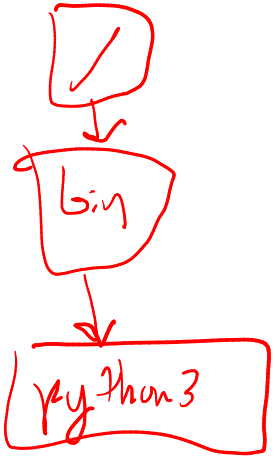
- If we are in "`/Users/chloe`", then "`.`" refers to "`/Users/chloe`" and "`..`" refers to "`/Users`"

`~` is your home directory (e.g. "/Users/chloe")

- The shell variable **HOME** contains the name of your home directory. You can see the value by echoing it: `echo $HOME`
- When used in place of a directory, a long tilde is expanded by the shell to the name of your home directory

echo ~

Global vs. Local Programs



wc

You can run a program by typing its path

- e.g. `/bin/python3 myscript.py`

Some folders are globally visible, so you only need the program's name

- /bin/ is globally visible because it is in the **PATH** shell variable (more on this later)
- e.g. `python3 myscript.py`

To run a program in the current directory, you need the path

- e.g. `./local_program`
- Running `local_program` by itself will not work, b/c it's not globally visible

File Manipulation

How do we affect the file system?

Common concepts across all operating systems

- See folder contents
- Go into folder
- Copy file
- Move/rename file
- Read file contents

Basic File Operations

Command	Operation	Example
`ls`	List files in a directory. The most important options are -a, -l, and -d.	`ls -l`
`cp <source> <destination>`	Make a copy of given file in given destination. Using the -a or -r option, you can also recursively copy directories.	`cp file.txt myDir/`
`mv <oldName> <newname>`	Rename or move given existing file to given name/destination	`mv fil.txt file.txt`
`rm <name>`	Delete or remove a file. -r option will recursively remove a directory and its contents.	`rm file.txt` `rm -r directory`

Directory Operations

Command	Operation	Example
`cd <directoryName>`	Change your current directory, move into the given directory.	`cd myDir` cd myDir/subdir
`pwd`	Prints the absolute path of your current working directory.	`pwd` cd ..
`mkdir <directoryName>`	Create an empty directory at location specified	`mkdir newDir`
`rmdir <directoryName>`	Delete (remove) an empty directory.	`rmdir newDir`
`rm -r <directoryName>`	Delete a nonempty directory and its contents. **Use with caution.**	`rm -r newDir`

File Viewing

Command	Operation	Example
`cat <fileName>`	Concatenate and print file contents to terminal window	`cat file.txt`
`less <filename>`	View text files one page at a time. Use less to view text one “page” at a time (i.e., one window or screenful at a time).	`less file.txt`
`nl <filename>`	View text files with their lines numbered.	`nl file.txt`
`head`	View the first lines of a text file. Default to first 10.	`head file.txt`
`tail`	View the last lines of a text file. Default to last 10.	`tail file.txt`

File Creation, Editing, and Properties

Command	Operation	Example
`emacs <fileName>`	Create or edit files using emacs, a text editor from Free Software Foudnation.	`emacs file.txt`
`vim <fileName>`	Create or edit files using vim, a text editor extended from Unix vi.	`vim file.txt`
`touch <filename>`	Create empty file.	`touch file.txt`
	nano filename	
`wc`	Count bytes, words, and lines in a file.	`wc file.txt`
`chmod`	Change the permissions mode of files and directories.	`chmod u+x script`

File Location and Text Manipulation

Command	Operation	Example
`find -name <filename>`	Search for file	`find -name file.txt`
`grep <pattern> <filename>`	Find lines in a file that match a regular expression.	`grep apple file.txt`
`sed <pattern> <filename>`	A pattern-matching engine that can perform manipulations on lines of text.	`sed 's/me/YOU/g' file.txt`
`sort <filename>`	Sort lines of text by various criteria.	`sort file.txt`
`uniq <filename>`	Locate identical lines in a file. uniq is often used after sorting a file.	`sort file.txt | uniq`

Demo: File Manipulation

Other Commands

Useful command: `grep`

Search for a given string within a given file

- **`grep [options] pattern [files]`**
- Example: `grep computer /usr/share/dict/words`

Helpful Options

- `-c` : prints count of lines with given pattern
- `-h` : display matched lines (without filenames)
- `-i` : ignore case when matching
- `-l` : display list of filenames with matches
- `-v`: invert match

```
[cf2024@calgary ~]$ grep computer /usr/share/dict/words
computer
computerese
computerise
computerite
computerizable
computerization
computerize
computerized
computerizes
computerizing
computerlike
computernik
computers
microcomputer
microcomputers
minicomputer
minicomputers
multicomputer
multimicrocomputer
supercomputer
supercomputers
telecomputer
```

Network Connections

ssh: securely log into a remote host, or run commands on it.

- `ssh user@remote.host`

scp: **s**ecure **c**opy. Uses `ssh` protocol to transfer files between different hosts

- `scp user@remote.host:file.txt /local/directory` copies `file.txt` from remote host to local directory
- `scp file.txt user@remote.host:/remote/directory/` copies `file.txt` from local host to remote directory
- Use `-r` option to recursively copy a remote directory to your local machine, or vice versa: `scp -r user@remote.host:mydir .`
    - Note the destination `.` – it refers to the current working directory

Web Browsing

wget: non-interactive download of files from the web supporting http, https and FTP

- Non interactive means it can work in the background (helpful if the files take a while)
- The wget command hits a URL and downloads the data to a file or standard output. It's great for capturing individual web pages, downloading files, or duplicating entire website hierarchies to arbitrary depth.
- `wget http://website.com/files/file.zip`

File Compression and Packaging

tar: tape archive. Compresses directory of files for easy transfer (like zip)

- **`tar -cvf <output> <directory to compress>`**
  - `tar -c -v -f myTarFile.tar /Users/chloe/`
  - `-c` – creates new `.tar` archive file
  - `-v` - Verbosely show the tar process
  - `-f` - to decide name of tar file
  - **Order matters!**
- **`tar -xvf <file to extract>`**
  - `tar -x -v -f myTarFile.tar`
  - Use `-C` to extract into a separate directory

Screen Output

`echo`: print simple text on standard output

- Example: **`echo Hello World`**

`clear`: clear the screen or window

Demo: Other Commands

Assignment Reminders

EX0: Course Policies due Friday 6/26 **@10:49 AM***
EX1: Shell due Friday 6/26 **@10:49 AM**

- *Exercises are normally due before the next lecture starts, but giving some extra time for EX0 as people are settling into class!

HW0: Shell Access due Friday 6/26 @11:59 PM

- Log into Calgary and experiment with the shell!

Pre-Course Survey due on Sunday 6/27 @11:59 PM

- We (the staff) want to get to know you better and how we can best support you!

Make sure to submit Poll: Lec 2 before tomorrow as well.

All of these assignments can be found on our course's Gradescope page.

Extra Notes