1. Open vscode
2. Go to ‘File –> Open workspace from file’
3. Go into your gen711 folder on the desktop
4. Select the workspace you saved last week.
5. Open terminal (menu bar –> terminal –> new terminal)
6. Determine if the terminal is in ‘gen711_lab’ folder

Today:

- Navigating Files and Directories (review and new)
- Practice practical question

Having touble changing directories in terminal to your lab folder? Try:

cd ~ cd gen711

Navigating Files and Directories

- How can I perform operations on files outside of my working directory?
- What are some navigational shortcuts I can use to make my work more efficient?

Objectives:

- Use a single command to navigate multiple steps in your directory structure, including moving backwards (one level up).
- Perform operations on files in directories outside your working directory.
- Work with hidden directories and hidden files.
- Interconvert between absolute and relative paths.
- Employ navigational shortcuts to move around your file system.

More navigation

- We got an idea for moving around using cd and the name of the folder to move into.
- But how to we go back out? We dont see the folder we are in.
- We have a special command to tell the computer to move us back or up one directory level.

We have a special command to tell the computer to move us back or up one directory level.

- we use ‘cd ..’

```
cd untrimmed_fastq
ls
```

- then, lets do the exact opposite to go back to where we were

```
cd ../
ls
>sra_metadata    untrimmed_fastq
```

- Navigation seems pretty slow, if you have to type cd every time if your data is like ten folders deep.
- Or the opposite is equally slow; navigating out one directory at a time with ‘../’
- lets go back into ‘untrimmed_fastq’

```
cd untrimmed_fastq
pwd
/home/unhAW/jtmiller/gen711/shell_data/untrimmed_fastq
```

- To navigate out two folders from here, we can double it up with

```
cd ../../
ls
>shell_data
```

- Use the tab to fugure out where you are

```
cd ../<tab>
cd ../../
```

Navigate by multiple directories

- Lets navigate to your home directory using '../'
  - if you are in gen711_lab, navigate one folder out. 'cd ../'
  - if you are in 'shell_data', navigate two folders out. 'cd ../../'
  - if you are in 'untrimmed_fastq' or '.hidden', navigate three directories out with 'cd ../../'

```
pwd
>/home/unhAW/jtmiller
```

The tilda shortcut takes you home

- If you end up in the wrong directory, or something that you do not recognize, remember that the (~, green/yellow/esc)

```
cd ~
```

- The /, ~, and .. characters represent important navigational shortcuts.
- Will be on the exam:
  - If the path starts with '/', it is an absolute path.
  - If the path starts with '../' or a names of a directory, it is a relative path.
- Relative paths specify the location starting from the current location,
- wWhile absolute paths specify a location from the root of the file system.

The / by itself, takes you all the way back to root.

- There are two differnt ways to change directories into shell data

- The relative path

```
cd gen711/shell_data
```

- *OR* the absolute path:

```
cd /home/unhAW/jtmiller/gen711/shell_data
```

- We have been navigating using relative paths
- But you can use absolute paths anywhere, and they will get you there
- Relative paths only work the directory that you want to navigate into is in your current working directory, or you can get there with '../'

Relative path

```
cd untrimmed_fastq
cd ../
```

Absolute

```
cd /Users/jeffreymiller/Desktop/gen711_lab/shell_data/untrimmed_fastq
```

Exercise 3a:

- Now, from 'untrimmed_fastq', what are three ways to navigate to your home directory?

Hidden files

- Hidden files and directories start with . and can be viewed using ls -a.
- Sometimes, its better to hide files from the command line. Like system files that shouldn't be messed with by the command line user.
- Somewhere in 'shell_data' is a hidden file.
- Mention ls -aF and ls -Fa are the same, but case matters alot

Working with Files and Directories

We are interested in looking at the FASTQ files in this directory. We can list all files with the .fastq extension using the command:

Questions How can I create, copy and delete files and directories? How can I control who has permission to modify a file? How can I repeat recently used commands?

Objectives View, search within, copy, move, and rename files. Create new directories. Use wildcards (*) to perform operations on multiple files. Make a file read only. Use the history command to view and repeat recently used commands.

Now that we know how to navigate around our directory structure, let's start working with our sequencing files. We did a sequencing experiment and have two results files, which are stored in our untrimmed_fastq directory.

```
cd ~/shell_data/untrimmed_fastq
```

- Lets say that we did some grepping here and made a new text file for an analysis.

```
grep '@' SRR097977.fastq > textfile.txt
ls
```

- Pipe is for making the output of one command go into another. The '>' command is directing the output to a file
  - People get tripped up on this. It will be on the exam.
- We might have many files in here, but we are omly interested in looking at the FASTQ files in this directory.
- We can list all files with the .fastq extension using the fastq part of the file, and something called a wildcard or glob:

```
 ls *.fastq
 > SRR097977.fastq  SRR098026.fastq
```

The wildcard works with more than just the file extension name. Similar to the way the tab works by filling in as much as it can without making decisions for you.

```
 ls *977.fastq
 > SRR097977.fastq
```

- Just for fun, lets do this with an absolute path as well. You can use this to see files that match in far directories too.

```
ls /Users/jeffreymiller/Desktop/gen711_lab/shell_data/untrimmed_fastq/*fastq
```

EXERCISE

1. Do each of the following tasks from your current directory using a single ls command for each:

    - List all of the files in /Applications that start with the letter ‘c’.
    - List all of the files in /Applications that contain the letter ‘a’.
    - List all of the files in /Applications that end with the letter ‘o’.
    - Bonus: List all of the files in /Applications that contain the letter ‘a’ or the letter ‘c’.
2. ‘echo’ is a built-in shell command that writes its arguments, like a line of text to standard output. The ‘echo’ command can also be used with pattern matching characters, such as wildcard characters. Here we will use the echo command to see how the wildcard character is interpreted by the shell. What does echo say when you try to match something that is not there? What about ls?

Don’t Scroll To Bottom! ANSWERS ARE BELOW

COMMAND HISTORY

- If you want to repeat a command that you’ve run recently, you can access previous commands using the up arrow on your keyboard to go back to the most recent command. This is helpful if you deleted something you needed. Lets intentionally remove our readinfo.txt, but then remake it from our history.
- Likewise, the down arrow takes you forward in the command history.
- A third way is to use the command ‘history’

To cancel something that wont run:

- ctrl + c (cancel command when stuck)
- ctrl + r (find in your command history)
- ctrl + L (clear)

Exercise:

- Find the line number in your history for the command that listed all the .fastq files using the absolute path . Rerun that command. history | grep -n ‘grep’

ANSWERS

1. ls /usr/bin/c _ls /usr/bin/_a _ls /usr/bin/_o Bonus: ls /usr/bin/_[ac]_
2. echo _.fastq > SRR097977.fastq SRR098026.fastq echo_ .missing ls *.missing

cd ~/shell_data/untrimmed_fastq

grep NNNNNNNNNN SRR098026.fastq

grep -B1 -A2 NNNNNNNNNN SRR098026.fastq

Exercise

Search for the sequence GNATNACCACTTCC in the SRR098026.fastq file. Have your search return all matching lines and the name (or identifier) for each sequence that contains a match. Search for the sequence AAGTT in both FASTQ files. Have your search return all matching lines and the name (or identifier) for each sequence that contains a match.