Lab introduction

Gemma Martínez Redondo & Karin Steffen

Many slides taken from Dr. Sophie Shaw
Daily workshop material, slides & exercises
http://evomics.org/2024-workshop-on-phylogenomics-cesky-krumlov/
Daily workshop material, slides & exercises
Daily workshop material, slides & exercises
Good workshop practice

• Work together
• Ask lots of questions
• Take breaks
• Use cheat sheets
• Have lots of fun
Unix/Linux Command Reference

Cheat sheets

<table>
<thead>
<tr>
<th>File Commands</th>
<th>System Info</th>
</tr>
</thead>
<tbody>
<tr>
<td>ls - directory listing</td>
<td>date - show the current date and time</td>
</tr>
<tr>
<td>la -al - formatted listing with hidden files</td>
<td>cal - show this month's calendar</td>
</tr>
<tr>
<td>cd directory</td>
<td>uptime - show current uptime</td>
</tr>
<tr>
<td>cd - change directory to dir</td>
<td>w - display who is online</td>
</tr>
<tr>
<td>pwd - show current directory</td>
<td>whoami - who you are logged in as</td>
</tr>
<tr>
<td>mkdir dir - create a directory dir</td>
<td>finger user - display information about user</td>
</tr>
<tr>
<td>rm file - delete file</td>
<td>uname -a - show kernel information</td>
</tr>
<tr>
<td>rm -r dir - delete directory dir</td>
<td>cat /proc/cpuinfo - cpu information</td>
</tr>
<tr>
<td>rm -rf dir - force remove file</td>
<td>cat /proc/meminfo - memory information</td>
</tr>
<tr>
<td>rm -rf dir - force remove directory dir *</td>
<td>man command - show the manual for command</td>
</tr>
<tr>
<td>cp file1 file2 - copy file1 to file2</td>
<td>df - show disk usage</td>
</tr>
<tr>
<td>cp -r dir1 dir2 - copy dir1 to dir2; create dir2 if it doesn't exist</td>
<td>du - show directory space usage</td>
</tr>
<tr>
<td>mv file1 file2 - rename or move file1 to file2</td>
<td>free - show memory and swap usage</td>
</tr>
<tr>
<td>if file2 is an existing directory, moves file1 into directory file2</td>
<td>whereis app - show possible locations of app</td>
</tr>
<tr>
<td>ln -s file link - create symbolic link to file</td>
<td>which app - show which app will be run by default</td>
</tr>
<tr>
<td>touch file - create or update file</td>
<td>Compression</td>
</tr>
<tr>
<td>cat &gt; file - places standard input into file</td>
<td>tar cf file.tar files - create a tar file</td>
</tr>
<tr>
<td>more file - output the contents of file</td>
<td>tar xf file.tar.gz files - create a tar file with gzip compression</td>
</tr>
<tr>
<td>head file - output the first 10 lines of file</td>
<td>tar xzf file.tar.gz - extract a tar file with gzip</td>
</tr>
<tr>
<td>tail file - output the last 10 lines of file</td>
<td>tar xjzf file.tar.bz2 - extract a tar file with bzip2</td>
</tr>
<tr>
<td>tail -f file - output the contents of file as it grows, starting with the last 10 lines</td>
<td>gzip -d file.gz - decompresses file.gz back to file</td>
</tr>
</tbody>
</table>

Process Management

| ps - display your currently active processes | ping host - ping host and output results |
| top - display all running processes | whois domain - get whois information for domain |
| kill pid - kill process id pid | dig domain - get DNS information for domain |
| killall proc - kill all processes named proc * | dig <host> - reverse lookup host |
| bg - lists stopped or background jobs, resume a stopped job in the background | wqf file - download file |
| fg - brings the most recent job to foreground | wqf -c file - continue a stopped download |
| fg n - brings job n to the foreground | INSTALLATION |

File Permissions

| chmod octal file - change the permissions of file to octal, which can be found separately for users, group, and world by adding: | Install from source: |
| 4 - read (r) | /configure |
| 2 - write (w) | make |
| 1 - execute (x) | make install |
| Examples: | dpkg -- -i pkg.deb - install a package (Debian) |
| chmod 777 - read, write, execute for all | rpm -oVh pkg.rpm - install a package (RPM) |
| chmod 755 - rwx for owner, rx for group and world | SHORTCUTS |
| For more options, see man chmod. | Ctrl+C - halts the current command |
| | Ctrl+Z - stops the current command, resume with fg in the foreground or bg in the background |
| | Ctrl+D - log out of current session, similar to exit |
| | Ctrl+W - erases one word in the current line |
| | Ctrl+0 - erases the whole line |
| | Ctrl+R - type to bring up a recent command |
| | ! - repeats the last command |
| | exit - log out of current session |

SSH

| ssh user@host - connect to host as user | * use with extreme caution. |
| ssh -p port user@host - connect to host on port port as user | |
| ssh-copy-id user@host - add your key to host for user to enable a key-based passwordless login | |

Search

<p>| grep pattern files - search for pattern in files | Command | grep pattern - search for pattern in the output of command |
| grep -r pattern dir - search recursively for pattern in dir | locate file - find all instances of file |</p>
<table>
<thead>
<tr>
<th>File Commands</th>
<th>System Info</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ls</code> - directory listing</td>
<td><code>date</code> - show current date and time</td>
</tr>
<tr>
<td><code>ls -al</code> - formatted listing with hidden files</td>
<td><code>cal</code> - show this month’s calendar</td>
</tr>
<tr>
<td><code>cd</code> - change directory to dir</td>
<td><code>uptime</code> - show current uptime</td>
</tr>
<tr>
<td><code>cd ~</code> - change directory to home</td>
<td><code>w</code> - display who is online</td>
</tr>
<tr>
<td><code>pwd</code> - show current directory</td>
<td><code>whoami</code> - who are you logged in as</td>
</tr>
<tr>
<td><code>mkdir</code> - create directory dir</td>
<td><code>finger</code> - display information about user</td>
</tr>
<tr>
<td><code>rm</code> - remove file</td>
<td><code>uname --a</code> - show kernel information</td>
</tr>
<tr>
<td><code>rm -rf</code> - remove directory dir</td>
<td><code>cat /proc/cpuinfo</code> - CPU information</td>
</tr>
<tr>
<td><code>rm -rf dir</code> - remove directory dir *</td>
<td><code>cat /proc/meminfo</code> - Memory information</td>
</tr>
<tr>
<td><code>cp file1 file2</code> - copy file1 to file2</td>
<td><code>man command</code> - show the manual for command</td>
</tr>
<tr>
<td><code>cp -r dir1 dir2</code> - copy dir1 to dir2, create dir2 if it doesn’t exist</td>
<td><code>df</code> - show disk usage</td>
</tr>
<tr>
<td><code>mv file1 file2</code> - rename or move file1 to file2 if file2 is an existing directory, moves file1 into directory file2</td>
<td><code>du</code> - show directory space usage</td>
</tr>
<tr>
<td><code>ln -s</code> - create symbolic link link to file</td>
<td><code>free</code> - show memory and swap usage</td>
</tr>
<tr>
<td><code>touch file</code> - create or update file</td>
<td><code>whereis app</code> - show possible locations of app</td>
</tr>
<tr>
<td><code>cd</code> - place the contents of file</td>
<td><code>which app</code> - show which app will be run by default</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Process Management</th>
<th>Compression</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ps</code> - display your currently active processes</td>
<td><code>tar</code> - create a tar file</td>
</tr>
<tr>
<td><code>top</code> - display all running processes</td>
<td><code>tar</code> - extract the files from tar file</td>
</tr>
<tr>
<td><code>kill pid</code> - kill process id</td>
<td><code>gzip</code> - compress file and name it to file.gz</td>
</tr>
<tr>
<td><code>killall proc</code> - kill all processes named proc *</td>
<td><code>gzip -d</code> - decompresses file.gz back to file</td>
</tr>
<tr>
<td><code>bg</code> - lists stopped or background jobs, resume a stopped job in the background</td>
<td><code>Network</code></td>
</tr>
<tr>
<td><code>fg</code> - brings the most recent job to foreground</td>
<td><code>ping host</code> - ping host and output results</td>
</tr>
<tr>
<td><code>fg n</code> - brings job n to the foreground</td>
<td><code>whois domain</code> - get WHOIS information for domain</td>
</tr>
<tr>
<td><code>chmod</code> - change the permissions of file to octal, which can be found separately for users, groups, and world by adding: 4 - read (r), 2 - write (w), 1 - execute (x)</td>
<td><code>dig domain</code> - get DNS information for domain</td>
</tr>
<tr>
<td>Examples: <code>chmod 777</code> - read, write, execute for all</td>
<td><code>dig @host</code> - reverse lookup host</td>
</tr>
<tr>
<td><code>chmod 755</code> - rwxs for owner, rx for group and world</td>
<td><code>wget</code> - download file</td>
</tr>
<tr>
<td>For more options, see man chmod.</td>
<td><code>wget -c</code> - continue a stopped download</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>File Permissions</th>
<th>Installation</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>chown</code> - change the ownership of file to octal, which can be found separately for users, groups, and world by adding: 4 - read (r), 2 - write (w), 1 - execute (x)</td>
<td><code>Install from source: ./configure make</code></td>
</tr>
<tr>
<td>Examples: <code>chown</code> - read, write, execute for all</td>
<td><code>make</code></td>
</tr>
<tr>
<td><code>chown 755</code> - rwxs for owner, rx for group and world</td>
<td><code>dpkg -i pkg.deb</code> - install a package (Debian)</td>
</tr>
<tr>
<td>For more options, see man chown.</td>
<td><code>rpm -Ovh pkg.rpm</code> - install a package (RPM)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SSH</th>
<th>Shortcuts</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ssh user@host</code> - connect to host as user</td>
<td>Ctrl+C - clears the current command</td>
</tr>
<tr>
<td><code>ssh -p port user@host</code> - connect to host on port as user</td>
<td>Ctrl+z - stops the current command, resume with fg in the foreground or bg in the background</td>
</tr>
<tr>
<td><code>ssh-copy-id user@host</code> - add your key to host for user to enable a keyed or passwordless login</td>
<td>Ctrl+d - log out of current session, similar to exit</td>
</tr>
<tr>
<td><code>curl +w</code> - erases one word in the current line</td>
<td>Ctrl+w - erases the whole line</td>
</tr>
<tr>
<td><code>curl +k</code> - type to bring up a recent command</td>
<td>Ctrl+x - log out of current session</td>
</tr>
<tr>
<td><code>curl +t</code> - repeats the last command</td>
<td><code>*</code> - use with extreme caution</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Searching</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><code>grep pattern files</code> - search for pattern in files</td>
<td><code>Shortcuts</code></td>
</tr>
<tr>
<td><code>grep -r pattern dir</code> - search recursively for pattern in dir</td>
<td>Ctrl+c - clears the current command</td>
</tr>
<tr>
<td>`command</td>
<td>grep pattern` - search for pattern in the output of command</td>
</tr>
<tr>
<td><code>locate file</code> - find all instances of file</td>
<td>Ctrl+d - log out of current session, similar to exit</td>
</tr>
<tr>
<td><code>locate command</code> - find all instances of command</td>
<td>Ctrl+w - erases the whole line</td>
</tr>
</tbody>
</table>

- `*` use with extreme caution
What is UNIX?

Operating system
What is UNIX?

Operating system

Why do we use it?

• Bioinformatics software designed to run on Unix platforms
• Large amounts of data
• Much faster than Windows PC
What is UNIX?
Operating system

Why do we use it?
• Bioinformatics software designed to run on Unix platforms
• Large amounts of data
• Much faster than Windows PC

... And how?
• Linux computers or servers
• Computer clusters
• The cloud
AWS availability zones
How it works

AMI
(“Amazon Machine Image”)

Base computer with all data and software
How it works

Own copy of AMI = Instance (Virtual machine, VM)
Terminology

• Creating an instance – buying a brand new computer with software already installed.

• Starting an instance – turning that computer on.

• Stopping an instance – turning that computer off.

• Terminating an instance – setting that computer on fire and throwing it out of the window.
Connecting to your instance

Windows: remote desktop software
Guacamole, X2GO

Linux/Mac: Terminal
SSH ("Secure shell")
Instance addresses

http://evomics.org/2024-workshop-on-phylogenomics-cesky-krumlov/

FACULTY AND GENOMICS

- Click here to see our Faculty arrival and departure dates
- Faculty Bios and Teaching Assistant Bios
- Faculty Lunches: sign-up sheet
  - Instance IPs
  - Cheatsheet – copying files from/to the instances
<table>
<thead>
<tr>
<th>Name</th>
<th>IP address</th>
<th>Guacamole connection</th>
<th>ssh connection</th>
<th>RStudio server connection</th>
<th>Username: phyloge</th>
</tr>
</thead>
<tbody>
<tr>
<td>Phylo-Karin</td>
<td>52.91.211.21</td>
<td>52.91.211.21:8080/guacamole</td>
<td>ssh genomics@52.91.211.21</td>
<td>52.91.211.21:8080/guacamole</td>
<td>Date:</td>
</tr>
<tr>
<td>Phylo-Marina</td>
<td>44.212.5.42</td>
<td>44.212.5.42:8080/guacamole</td>
<td>ssh genomics@44.212.5.42</td>
<td>44.212.5.42:8080/guacamole</td>
<td></td>
</tr>
<tr>
<td>Phylo-Jacob</td>
<td>3.89.92.242</td>
<td>3.89.92.242:8080/guacamole</td>
<td>ssh genomics@3.89.92.242</td>
<td>3.89.92.242:8080/guacamole</td>
<td></td>
</tr>
<tr>
<td>Phylo-Gemma</td>
<td>54.205.120.236</td>
<td>54.205.120.236:8080/guacamole</td>
<td>ssh genomics@54.205.120.236</td>
<td>54.205.120.236:8080/guacamole</td>
<td></td>
</tr>
<tr>
<td>Phylo-Michal</td>
<td>44.207.6.114</td>
<td>44.207.6.114:8080/guacamole</td>
<td>ssh genomics@44.207.6.114</td>
<td>44.207.6.114:8080/guacamole</td>
<td></td>
</tr>
<tr>
<td>Phylo-Rosa</td>
<td>3.94.82.239</td>
<td>3.94.82.239:8080/guacamole</td>
<td>ssh genomics@3.94.82.239</td>
<td>3.94.82.239:8080/guacamole</td>
<td></td>
</tr>
</tbody>
</table>

Find your name and copy your IP address.
• Open your internet browser (e.g. Google Chrome)
• Paste the IP address followed by ‘:8080/guacamole’
• 52.91.211.21:8080/guacamole
Enter the username “phylogenomics” and password
Select Desktop, enter the same user name and pw again
Open terminal window using this icon

You’re now connected and you’re ready to learn some Unix!
But First...

- The domain address will change every day after we stop and re-start the instances.
- Each morning, you will need to return to the “Instance List” webpage, retrieve your new address and log in again.
Copy & Paste

AVOID COPYING AND PASTING WHEREVER POSSIBLE!
But if you do need to...

Press Ctrl+Alt+Shift

Paste the text into the box with right click → Paste

Press Ctrl+Alt+Shift again
You can now paste into the instance using right click
Your final task before we get started!

Make sure that typing tilde (~), backslash (\), pipe (|), and carat (^) in the terminal works.

Google search to find these on your computer if you don’t know where they are.
Questions?
## Commands overview

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>pwd</code></td>
<td>print working directory</td>
</tr>
<tr>
<td><code>~</code></td>
<td>home</td>
</tr>
<tr>
<td><code>.</code></td>
<td>here</td>
</tr>
<tr>
<td><code>..</code></td>
<td>one directory up</td>
</tr>
<tr>
<td><code>mkdir</code></td>
<td>make new directory</td>
</tr>
<tr>
<td><code>cd</code></td>
<td>change directory</td>
</tr>
<tr>
<td><code>touch</code></td>
<td>create file</td>
</tr>
<tr>
<td><code>ls</code></td>
<td>list</td>
</tr>
<tr>
<td><code>man</code></td>
<td>manual</td>
</tr>
<tr>
<td><code>mv</code></td>
<td>move</td>
</tr>
<tr>
<td><code>rm</code></td>
<td>remove</td>
</tr>
<tr>
<td><code>cp</code></td>
<td>copy</td>
</tr>
<tr>
<td><code>gunzip</code></td>
<td>unzip</td>
</tr>
<tr>
<td><code>tar</code></td>
<td>unarchive</td>
</tr>
<tr>
<td><code>head</code></td>
<td>first (n=10) lines</td>
</tr>
<tr>
<td><code>tail</code></td>
<td>last (n=10) lines</td>
</tr>
<tr>
<td><code>cat</code></td>
<td>concatenate</td>
</tr>
<tr>
<td><code>wc</code></td>
<td>word count</td>
</tr>
<tr>
<td><code>grep</code></td>
<td>pattern search</td>
</tr>
<tr>
<td>`</td>
<td>`</td>
</tr>
<tr>
<td><code>sed</code></td>
<td>stream editor</td>
</tr>
<tr>
<td><code>chmod</code></td>
<td>change file modes</td>
</tr>
</tbody>
</table>
The terminal (command line, shell, prompt)

Where you see this “$” followed by text, I want you to type the text on your command line
Location is important

First task: Where am I?

```
$ pwd
```

This is your working directory, i.e. where you currently are.
Creating and navigating directories
Creating and navigating directories

- **home directory**
  - ~/
  - ~
  - phylogenomics
  - master
  - ubuntu
- **root**
Creating and navigating directories

home directory
~

phylogenomics
~

master
ubuntu

bin
lib

"root"
Creating and navigating directories

Create a new directory called “Data” in your current directory

$ mkdir ./Data

Change into the new directory

$ cd Data

Where are you now? What is your present working directory?

Directory names (and file names for the matter) can not contain spaces.*
Underscores are often used instead if you want to separate words.
Creating and navigating directories

```
/    home    bin    lib
     |       |       |
     |       |       |
phylogenomics master ubuntu
```

```
Data
```
Creating and navigating directories

Make an empty file “rags”
$ touch rags

And another two “Heaven” and “Earth”
$ touch Heaven Earth

Now let’s list the contents of the current directory (Data)
$ ls
Creating and navigating directories

Now list all of the files in the directory

$ ls -a
Creating and navigating directories

Now list all of the files in the directory

$ ls -a
Creating and navigating directories

Now list **all** of the files in the directory

```
$ ls -a
```

. points to the current directory

.. points to one directory above
Creating and navigating directories

Now list all of the files in the directory

```
$ ls -a
```

. points to the current directory

.. points to one directory above
Creating and navigating directories

. and .. are used for specifying location

Whenever you do anything on Unix (move around, move a file, rename a file, run a program or script, etc...) you have to tell the system where that thing is using a path.

. and .. are part of RELATIVE paths
Creating and navigating directories

Create a directory called New within the phylogenomics directory using the RELATIVE PATH

$ mkdir ../New
Creating and navigating directories

Move from Data to New
RELATIVE PATH

$ cd ..:/New
Creating and navigating directories

Move from New to home
RELATIVE PATH

$ cd ..../../
Creating and navigating directories

Relative paths will always change depending on your location.
The alternative is ABSOLUTE paths.
These always start from root and will never change.
Creating and navigating directories

Move from your home directory to New

$ cd /home/phylogenomics/New
Creating and navigating directories

Move from your home directory to New
ABSOLUTE PATH

```
$ cd /home/phylogenomics/New
```

Move from Data to New
ABSOLUTE PATH

```
$ cd /home/phylogenomics/New
```
A note about . dot

. means in your (present) working directory

This command means “List everything that’s in the (present) working directory”

$ ls ./

This command means “List everything that’s in the working directory within a subdirectory called Data”

$ ls ./Data/

In most cases, people don’t use ./ at the beginning of a path. As long as the file/directory is within your working directory, the command will work.

$ ls ./Data/ = $ ls Data
Let’s practice

Where am I right now? (Should be the Data directory) $ pwd
Change to the directory above $ cd ../
Let’s list the contents of the Data directory $ ls ./Data

CHALLENGE 1!

1. Move into the Data directory and list the contents of your home directory.
2. In Data, make a new directory and move into this location.
3. From this new directory, move into your home directory IN ONE COMMAND and check your location.
Work smarter, not harder!

Tab completion is a nice trick to save you typing paths.

For this example we are going to list everything in the directory `/home/phylogenomics/workshop_materials/`. Start by typing `$ ls /` followed by pressing tab twice quickly. This shows the contents of the root directory:
Work smarter, not harder!

Now type: `$ ls /h` followed by tab once. The path to the `/home/` directory has filled in.

Now type: `$ ls /home/p` followed by tab once. The path to the `/home/phylogenomics/` directory has filled in.

Finally type: `$ ls /home/phylogenomics/w` followed by tab once to finish the path, and then enter.

You’ve now listed that directory contents.

Tab complete will fill in paths, save you time in typing and prevent typos!
Work smarter, not harder!

Two more tricks for less typing!
The * (asterisk) represents any character
For example: $ ls /home/phylogenomics/*.*.txt
Will list everything in my home directory ending .txt

The up arrow can be used to re-run commands
Press your up arrow and see!
If you want all of your previous commands listed, simply type $ history
Questions?
Binary programs

These are all programs installed on the Unix machine. They can be found in /bin.

$ ls /bin

These include `pwd`, `mkdir`, `ls` ...
Binary programs have manuals

To view the manual page, type man followed by the name of the program.

Open the manual page for ls

\$ man ls

Scroll through (enter) and find the options for:

- long listing format (-l),
- human-readable file sizes (-h),
- sort by modification time (-t).

Exit the manual page (type q) and give these ls options a go in your Data directory.

\$ ls -l -t -h ./Data = \$ ls -lth ./Data
PATH

The computer needs to know where a program is so that it can access the code to run the program.

The PATH environment variable is a list of locations your computer looks for programs.

You can either provide the path to the program you want to run

$ /usr/bin/mkdir
The computer needs to know where a program is so that it can access the code to run the program.

The PATH environment variable is a list of locations your computer looks for programs.

You can either provide the path to the program you want to run

$ /usr/bin/mkdir

Or make sure the program is in your PATH environment variable

To view locations in your PATH environment variable: $ echo $PATH

There are ways to add new locations to your PATH, but that is for another time.
Let’s practice some more

First I need you to make a new directory called “Working” within your home directory.

Afterwards your file structure should look like this.

Earth    Heaven     rags
Let’s practice some more

Moving Files

Let's move Heaven and Earth from Data to Working

```bash
$ cd ~/Data
$ mv ./Heaven ../Working
```

Now move Earth, too.
Remember to Tab complete!

file path
destination path
Let’s practice some more

Moving Files
mv can also be used to rename files.
Let’s change rags to riches.

```
$ mv ./rags ./riches
```

file path          destination path
~/phylogenomics    riches
|                 | Earth    Heaven
|                 | Working  |
|                 | Data     |
| home            | bin      | lib
|                 | main     | ubuntu
| phylogenomics   | main     | ubuntu
|                 | Data     | Working
|                 | riches   | Earth    Heaven
|                 |          |          |
Let’s practice some more

Deleting Files
Now let’s delete Heaven
(Check your present working directory is Data)

```
$ rm -i ../Working/Heaven
```

When prompted type y for yes and press enter.
Let’s practice some more

Deleting Files

Now let’s delete the entire ‘Working’ directory including Earth.

```
$ rm -i ../Working/Heaven
```
Let’s practice some more

Deleting Files
There is no ‘Trash’ or ‘Recycle Bin’ in Unix!
Once gone, files are gone forever!
Therefore try to ALWAYS use `rm -i`
Let’s practice some more

Let’s make a copy of riches within the home directory.

(Make sure your present working directory is Data.)

```
$ cp ./*.riches ../riches
```

**Copying Files**

file path  destination path
Let’s practice some more

Copying Files

You can also copy entire directories and use this function to rename files/directories.

Move to home  

Make a copy of the Data directory here and call it Backup.

$ cp -r ./Data ./Backup
Data management

Some files can become quite big so people will archive directories and compress large files so that they are easier to store or share. Here’s an example: sequences.tar.gz

.tar – means that it is a tape archived directory
.gz – means that it is gzipped file

These can be used alone or in combination

To uncompressed a tar archive (x = extract, v = verbose, f = all files)

$ tar -xvf <filename>

A Gzipped file

$ gunzip <filename>

A Gzipped Tar archive

$ tar -xzvf <filename>
Challenge 2

1. Change to the workshop_materials directory at the following path: 
`~/workshop_materials/unix`
You should find a compressed directory: `Sequences.tar.gz`
2. Make a copy of this file in the Backup directory you created earlier
3. Unarchive the original directory
4. Unzip the read files
5. Rename the unarchived files – `sequence_1.fq` and `sequence_2.fq`
6. Delete the original .tar file

```
tar gunzip cp mv rm -i cd gunzip mv mkdir
```
Questions?
**Viewing and manipulating files**

Navigate to the `workshop_materials` directory:

```
$ cd ~/workshop_materials
```

Unarchive the `Blast_Out.tar.gz`:

```
$ tar -xzvf Blast_Out.tar.gz
```
Viewing and manipulating files

phylogenomics

workshop_materials

CP_Blast_seqs.fna LP_Blast_seqs.fna

LINUX TERMINAL FOR BEGINNERS
Viewing and manipulating files

```
phylogenomics

workshop_materials

View the first 10 lines of a file
$ head CP_Blast_seqs.fna

To view the first 30 lines of the file
$ head -n 30 CP_Blast_seqs.fna
```
Viewing and manipulating files

View the last 10 lines of a file

```
$ tail CP_Blast_seqs.fna
```

To view the last 30 lines of the file

```
$ tail -n 30 CP_Blast_seqs.fna
```
Viewing and manipulating files

Print the entire file
$ cat CP_Blast_seqs.fna

Many files are too large to meaningfully view in terminal or to edit in a unix text editor.
Viewing and manipulating files

phylo-genomics

workshop_materials

CP_Blast_seqs.fna
LP_Blast_seqs.fna
All_blast_seqs.fna

‘cat’ can also combine multiple files

```
$ cat CP_Blast_seqs.fna LP_Blast_seqs.fna > All_blast_seqs.fna
```

And then count the number of lines in each fasta file to confirm they have been combined.

```
$ wc -l *.fna
```
Viewing and manipulating files

Use `grep` to print occurrences of a pattern

```
$ grep "\>" CP_Blast_seqs.fna
```
Viewing and manipulating files

Use ‘grep’ to print occurrences of a pattern

$ grep “>” CP_Blast_seqs.fna

Create a new files of the fasta headers

$ grep “>” CP_Blast_seqs.fna > CP_blast_headers.txt

$ grep “>” LP_Blast_seqs.fna > LP_blast_headers.txt
Viewing and manipulating files

Use `grep` for to count the number of times pattern occurs

```
$ grep -c ">" CP_Blast_seq.s.fna
```
Viewing and manipulating files

Use ‘grep’ for to count the number of times pattern occurs

$ grep -c “>” CP_Blast_seqs.fna

Quotations marks are vital!

$ grep -c > CP_Blast_seqs.fna

CP_Blast_seqs.fna is now empty and this can’t be undone.
Viewing and manipulating files

What can you do to your file to protect them? Change permissions

$ chmod 444 CP_Blast_seq.ses.fna
Viewing and manipulating files

What can you do to your file to protect them? Change permissions

```bash
$ chmod 444 CP_Blast_seqs.fna
```

You might know and need this when you write scripts you want to execute. Sometimes you need to change permission to allow them to be executed.

```bash
$ chmod +x my_script.sh
```
Viewing and manipulating files

Search for headers that are not partial sequences

```
$ grep -v "partial" LP_headers.txt
```
Viewing and manipulating files

Replacing Text in Large files

```
sed 's/**FIND**/**REPLACE**/g' filename > output_file
```

```
$ sed "s/ _/g" CP_blast_headers.txt
```

phylogenomics

workshop_materials

CP_Blast_seqs.fna
LP_Blast_seqs.fna
All_blast_seqs.fna
CP_blast_headers.txt
LP_blast_headers.txt
Viewing and manipulating files

Replacing Text in Large files

```
sed 's/**FIND**/**REPLACE**/g' filename > output_file

$ sed "s/ /_/g" CP_blast_headers.txt
```

Use the -i flag to make changes in place

```
$ sed -i "s//,//g" CP_blast_headers.txt
$ head CP_blast_headers.txt
```
## Regular expressions

<table>
<thead>
<tr>
<th>Expression</th>
<th>Modern equivalent</th>
<th>Pattern matched</th>
</tr>
</thead>
<tbody>
<tr>
<td>.</td>
<td></td>
<td>a single character</td>
</tr>
<tr>
<td>.+</td>
<td></td>
<td>one or more characters</td>
</tr>
<tr>
<td>.*</td>
<td></td>
<td>zero or more characters</td>
</tr>
<tr>
<td>.?</td>
<td></td>
<td>maybe present</td>
</tr>
<tr>
<td>^</td>
<td></td>
<td>first in the line</td>
</tr>
<tr>
<td>$</td>
<td></td>
<td>last in the line</td>
</tr>
<tr>
<td>[0-9]</td>
<td>\d</td>
<td>digits</td>
</tr>
<tr>
<td>[a-zA-Z]</td>
<td>\w</td>
<td>letters</td>
</tr>
<tr>
<td>''</td>
<td>\s \t</td>
<td>space</td>
</tr>
<tr>
<td>{2}</td>
<td></td>
<td>exactly 2 characters long</td>
</tr>
<tr>
<td>{2,4}</td>
<td></td>
<td>between 2 and 4 characters long</td>
</tr>
<tr>
<td>[ACGT]</td>
<td></td>
<td>a specific set of characters</td>
</tr>
</tbody>
</table>
Regular expressions

Use sed -E to use extended regular expressions

$ sed -E 's/(>[A-Z0-9.]+)(.+)/\1/' CP_blast_headers.txt

Store pattern in memory using parentheses
Print out only the GenBank accessions
Regular expressions

Use sed -E to use extended regular expressions

$ sed -E 's/([A-Z0-9.]+)(.+)/\2/' CP_blast_headers.txt

Store pattern in memory using parentheses
Print out the rest of the headers
Pipes

Combine the fasta files from the blast output and identify how many sequences they contain

$ cat CP_Blast_seqs.fna LP_Blast_seqs.fna | grep ">" | wc -l

When building pipelines, it is useful to pipe to head to follow along

$ cat CP_Blast_seqs.fna LP_Blast_seqs.fna | head
$ cat CP_Blast_seqs.fna LP_Blast_seqs.fna | grep ">" | head
$ cat CP_Blast_seqs.fna LP_Blast_seqs.fna | grep ">" | wc -l
Pipes

String together many commands to count the number of unique accessions from these blast results

$ cat *.{fna}  
Print out all fasta files

$ cat *.{fna} | grep ">"  
Find all the fasta headers

Extract accessions

$ cat *.{fna} | grep ">" | sed -E "s/(>[A-Z0-9.]+)(.+)/\1/"  
Sort accessions

$ cat *.{fna} | grep ">" | sed -E "s/(>[A-Z0-9.]+)(.+)/\1/" | sort

Keep only unique accessions

$ cat *.{fna} | grep ">" | sed -E "s/(>[A-Z0-9.]+)(.+)/\1/" | sort | uniq

Count unique accessions

$ cat *.{fna} | grep ">" | sed -E "s/(>[A-Z0-9.]+)(.+)/\1/" | sort | uniq | wc -l
Challenge 3

Imagine you want to obtain the GenBank Accessions from the headers and you want to remove the version. Why is this command not working as expected?

```
$ sed -E "s/(>[A-Z0-9.]+)(.+)/\1/" CP_blast_headers.txt | sed "s/.1//g"
```

What would you need to change to make it work?
Loops

Iterate over e.g. files to execute a command repeatedly.

```bash
$ for i in *; do echo $i; done
```

Assign variable “i” to all files that end in .fna, then carry out command on all values of “i”.

```bash
$ for i in *.fna; do grep -c "CAT" $i; done
```

There is usually more than one way to do things. Try the following:

```bash
$ grep -c "CAT" *.fna
```
Downloading

From an internet URL
wget <url>
wget -P /path/to/where/the/download/should/be <url>

Save your data every day
We will launch new instance daily so everything you have done today will be gone tomorrow. Use scp or rsync to copy the notes you want to save to your own computer,

scp -r phylogenomics@<your.IP.address>:/Location/On/Instance /Local/path
rsync -avz phylogenomics@<your.IP.address>:/Location/On/Instance /Local/path

$ rsync -avz phylogenomics@ec ... .
Downloading

GUI: Filezilla [https://sourceforge.net/projects/filezilla/](https://sourceforge.net/projects/filezilla/)

Instance address
Username
Password

![Filezilla GUI screenshot](image-url)
Logging in to the instance through your local terminal

```
ssh -Y phylogenomics@<your.IP.address>
```

Accept and continue the connection with ‘yes’
Questions?
Quests!

Do these quests in whichever order you like. Whatever sounds interesting to you.

Easy

• open, modify and exit with saving a text file in an editor of your choice: nano, vim, emacs
• explore the commands: tr, cut, less, tree
• How do you append to a file without overwriting it?

You haven’t heard about all of this. Use your creativity, neighbor and the internet to tackle these!

Fun

Beautiful figures: Tidy Tuesday, R graph gallery, xkcd style, color choices (Ghibli, Wes Anderson, I want hue, Scico)

Challenging

• what about the programming language ‘awk’? :P

What are your favorite one-liners or UNIX tricks? (Collect them in a Google Docs!)