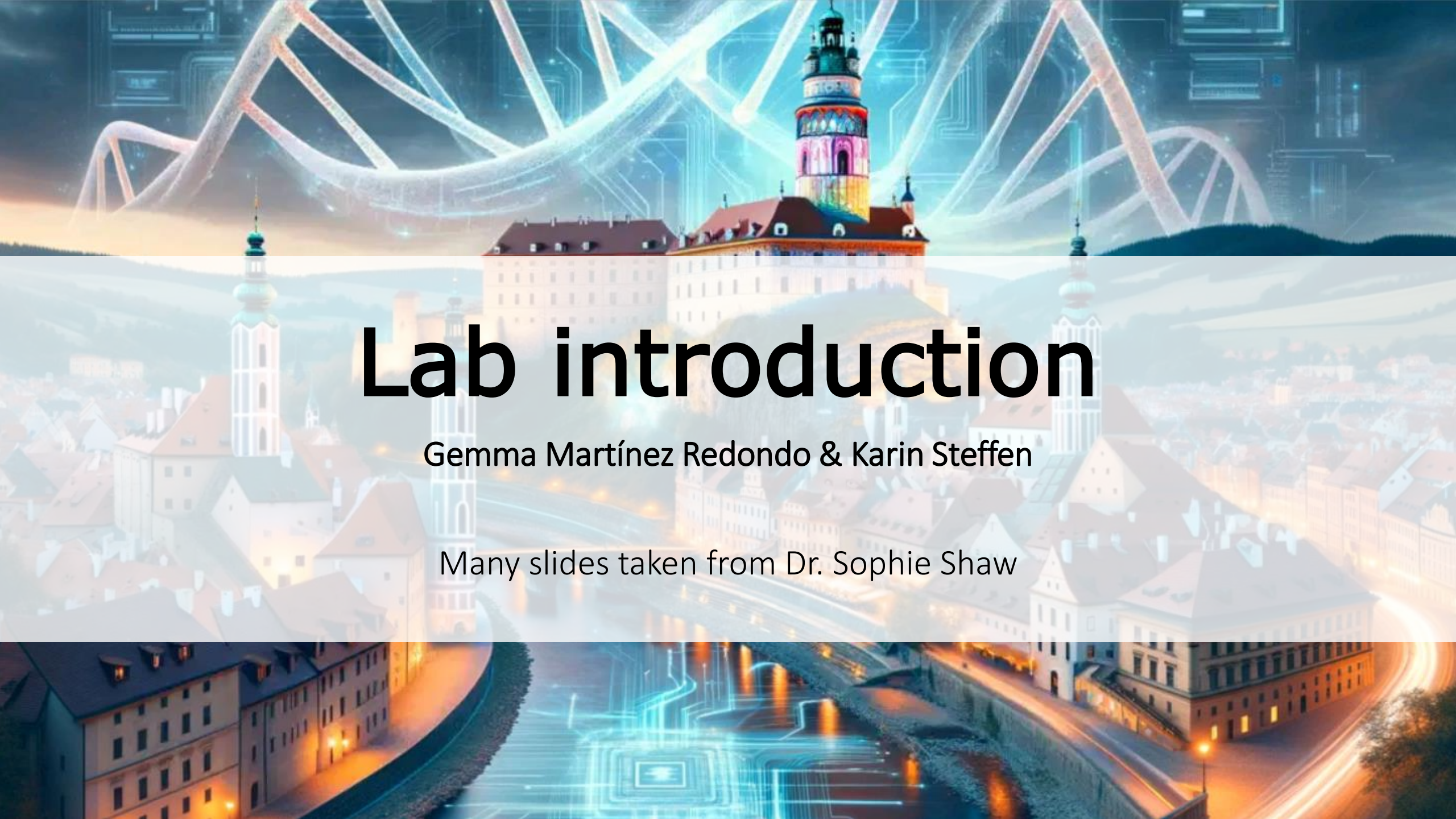


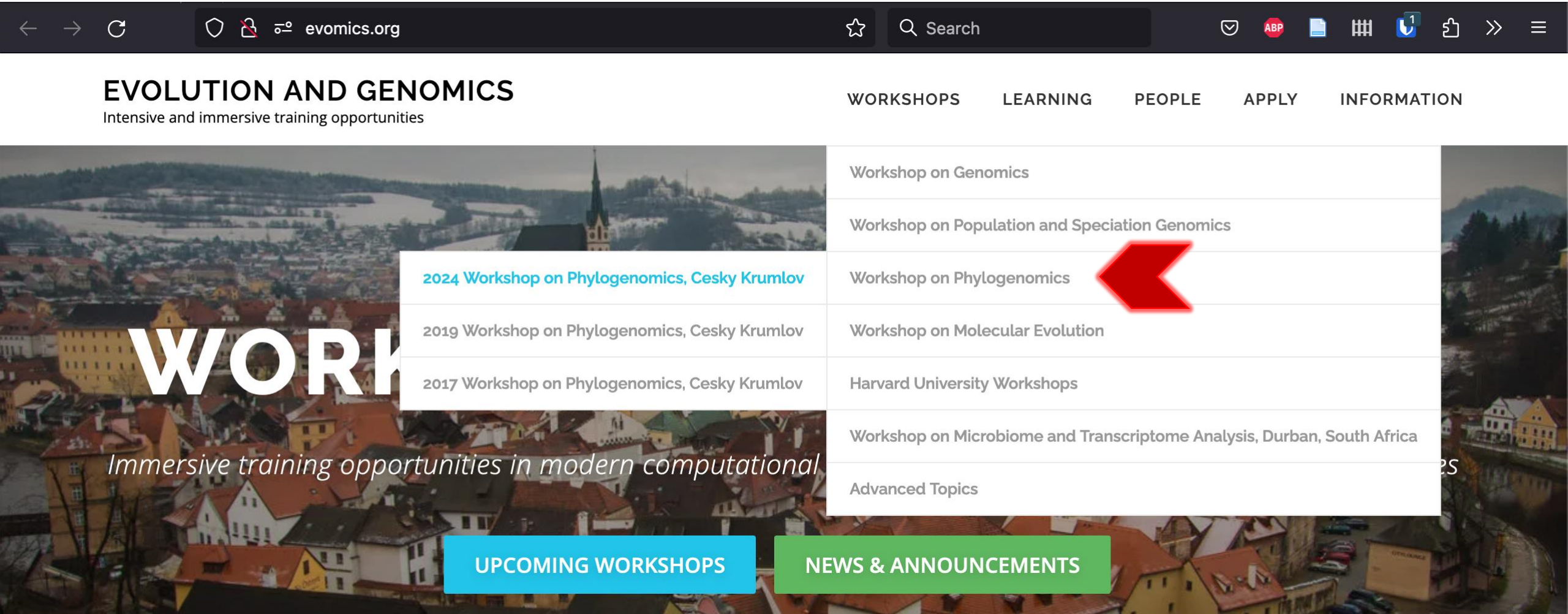
Lab introduction

Gemma Martínez Redondo & Karin Steffen

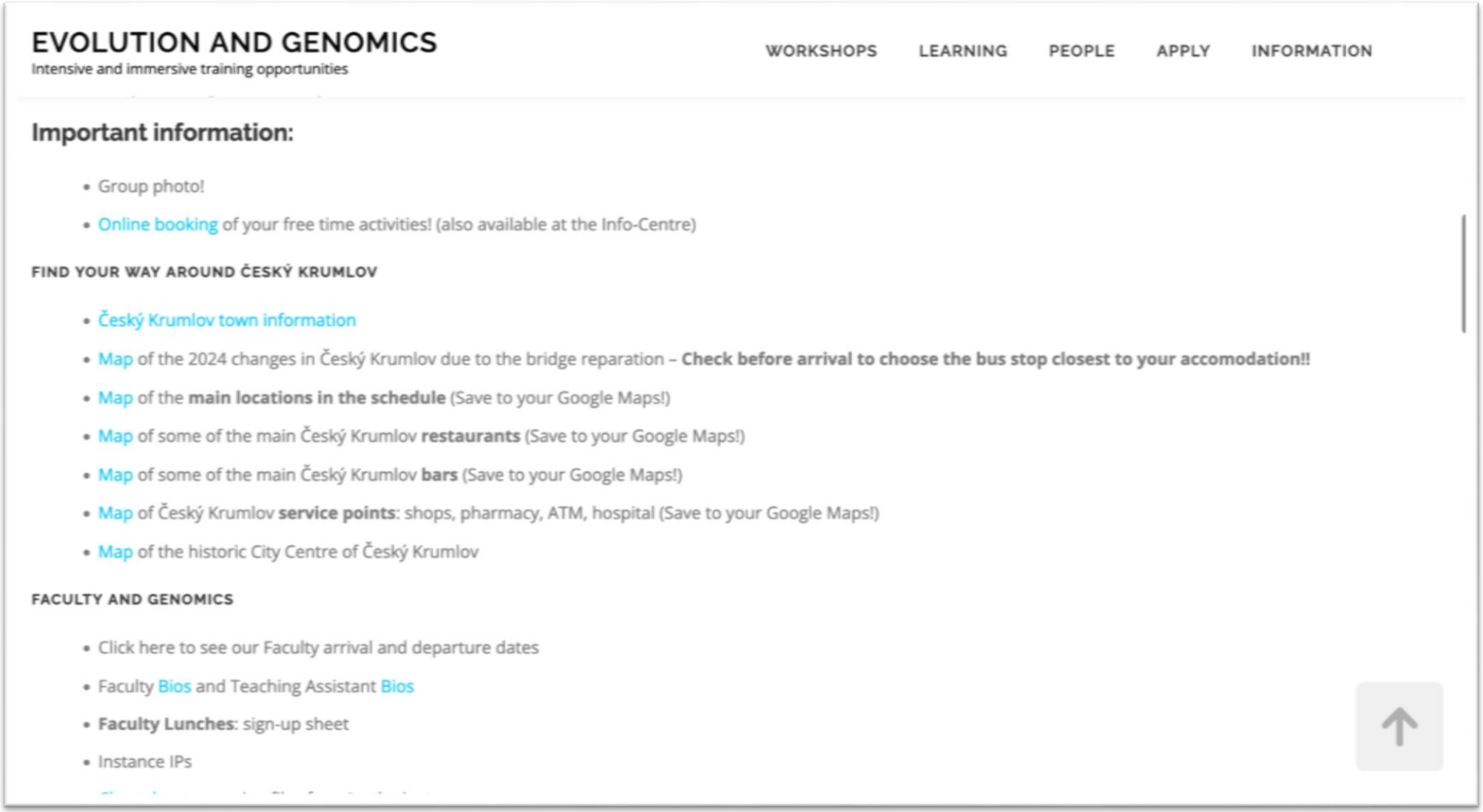
Many slides taken from Dr. Sophie Shaw

Daily workshop material, slides & exercises

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



Daily workshop material, slides & exercises



EVOLUTION AND GENOMICS
Intensive and immersive training opportunities

WORKSHOPS LEARNING PEOPLE APPLY INFORMATION

Important information:

- Group photo!
- Online booking of your free time activities! (also available at the Info-Centre)

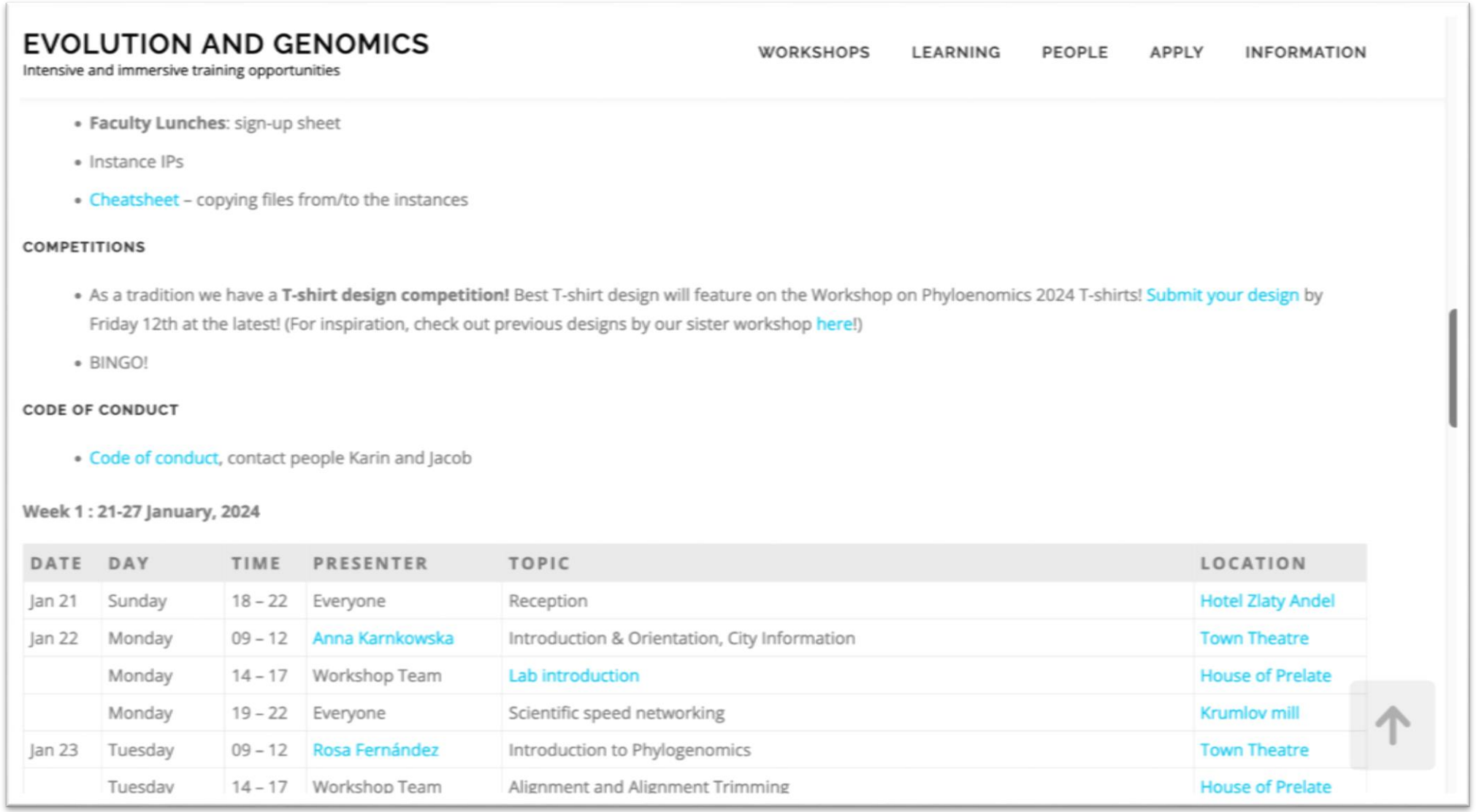
FIND YOUR WAY AROUND ČESKÝ KRUMLOV

- Český Krumlov town information
- Map of the 2024 changes in Český Krumlov due to the bridge reparation – **Check before arrival to choose the bus stop closest to your accomodation!!**
- Map of the **main locations in the schedule** (Save to your Google Maps!)
- Map of some of the main Český Krumlov **restaurants** (Save to your Google Maps!)
- Map of some of the main Český Krumlov **bars** (Save to your Google Maps!)
- Map of Český Krumlov **service points**: shops, pharmacy, ATM, hospital (Save to your Google Maps!)
- Map of the historic City Centre of Český 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

Daily workshop material, slides & exercises



EVOLUTION AND GENOMICS
Intensive and immersive training opportunities

WORKSHOPS LEARNING PEOPLE APPLY INFORMATION

- **Faculty Lunches**: sign-up sheet
- Instance IPs
- Cheatsheet – copying files from/to the instances

COMPETITIONS

- As a tradition we have a **T-shirt design competition!** Best T-shirt design will feature on the Workshop on Phyloenomics 2024 T-shirts! Submit your design by Friday 12th at the latest! (For inspiration, check out previous designs by our sister workshop here!)
- BINGO!

CODE OF CONDUCT

- Code of conduct, contact people Karin and Jacob

Week 1 : 21-27 January, 2024

DATE	DAY	TIME	PRESENTER	TOPIC	LOCATION
Jan 21	Sunday	18 – 22	Everyone	Reception	Hotel Zlaty Andel
Jan 22	Monday	09 – 12	Anna Karnkowska	Introduction & Orientation, City Information	Town Theatre
	Monday	14 – 17	Workshop Team	Lab introduction	House of Prelate
	Monday	19 – 22	Everyone	Scientific speed networking	Krumlov mill
Jan 23	Tuesday	09 – 12	Rosa Fernández	Introduction to Phylogenomics	Town Theatre
	Tuesday	14 – 17	Workshop Team	Alignment and Alignment Trimming	House of Prelate

Good workshop practice

- Work together
- Ask lots of questions
- Take breaks
- Use cheat sheets
- Have lots of fun

Cheat sheets

Unix/Linux Command Reference

FOSSwire.com

File Commands

ls – directory listing
ls -al – formatted listing with hidden files
cd *dir* - change directory to *dir*
cd – change to home
pwd – show current directory
mkdir *dir* – create a directory *dir*
rm *file* – delete *file*
rm -r *dir* – delete directory *dir*
rm -f *file* – force remove *file*
rm -rf *dir* – force remove directory *dir* *
cp *file1 file2* – copy *file1* to *file2*
cp -r *dir1 dir2* – copy *dir1* to *dir2*; create *dir2* if it doesn't exist
mv *file1 file2* – rename or move *file1* to *file2*
if *file2* is an existing directory, moves *file1* into directory *file2*
ln -s *file link* – create symbolic link *link* to *file*
touch *file* – create or update *file*
cat > *file* – places standard input into *file*
more *file* – output the contents of *file*
head *file* – output the first 10 lines of *file*
tail *file* – output the last 10 lines of *file*
tail -f *file* – output the contents of *file* as it grows, starting with the last 10 lines

Process Management

ps – display your currently active processes
top – display all running processes
kill *pid* – kill process id *pid*
killall *proc* – kill all processes named *proc* *
bg – lists stopped or background jobs; resume a stopped job in the background
fg – brings the most recent job to foreground
fg *n* – brings job *n* to the foreground

File Permissions

chmod *octal file* – change the permissions of *file* to *octal*, which can be found separately for user, group, and world by adding:

- 4 – read (r)
- 2 – write (w)
- 1 – execute (x)

Examples:
chmod 777 – read, write, execute for all
chmod 755 – rwx for owner, rx for group and world
For more options, see **man chmod**.

SSH

ssh *user@host* – connect to *host* as *user*
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 keyed or passwordless login

Searching

grep *pattern files* – search for *pattern* in *files*
grep -r *pattern dir* – search recursively for *pattern* in *dir*
command* | grep *pattern – search for *pattern* in the output of *command*
locate *file* – find all instances of *file*

System Info

date – show the current date and time
cal – show this month's calendar
uptime – show current uptime
w – display who is online
whoami – who you are logged in as
finger *user* – display information about *user*
uname -a – show kernel information
cat /proc/cpuinfo – cpu information
cat /proc/meminfo – memory information
man *command* – show the manual for *command*
df – show disk usage
du – show directory space usage
free – show memory and swap usage
whereis *app* – show possible locations of *app*
which *app* – show which *app* will be run by default

Compression

tar cf *file.tar files* – create a tar named *file.tar* containing *files*
tar xf *file.tar* – extract the files from *file.tar*
tar czf *file.tar.gz files* – create a tar with Gzip compression
tar xzf *file.tar.gz* – extract a tar using Gzip
tar cjf *file.tar.bz2* – create a tar with Bzip2 compression
tar xjf *file.tar.bz2* – extract a tar using Bzip2
gzip *file* – compresses *file* and renames it to *file.gz*
gzip -d *file.gz* – decompresses *file.gz* back to *file*

Network

ping *host* – ping *host* and output results
whois *domain* – get whois information for *domain*
dig *domain* – get DNS information for *domain*
dig -x *host* – reverse lookup *host*
wget *file* – download *file*
wget -c *file* – continue a stopped download

Installation

Install from source:
./configure
make
make install
dpkg -i *pkg.deb* – install a package (Debian)
rpm -Uvh *pkg.rpm* – install a package (RPM)

Shortcuts

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+U – erases the whole line
Ctrl+R – type to bring up a recent command
!! - repeats the last command
exit – log out of current session

* use with extreme caution.

Cheat sheets

Stack overflow, ChatGTP, Bard (Google).

Unix/Linux Command Reference

FOSSwire.com

File Commands

ls – directory listing
ls -al – formatted listing with hidden files
cd *dir* - change directory to *dir*
cd – change to home
pwd – show current directory
mkdir *dir* – create a directory *dir*
rm *file* – delete *file*
rm -r *dir* – delete directory *dir*
rm -f *file* – force remove *file*
rm -rf *dir* – force remove directory *dir* *
cp *file1 file2* – copy *file1* to *file2*
cp -r *dir1 dir2* – copy *dir1* to *dir2*; create *dir2* if it doesn't exist
mv *file1 file2* – rename or move *file1* to *file2* if *file2* is an existing directory, moves *file1* into directory *file2*
ln -s *file link* – create symbolic link *link* to *file*
touch *file* – create or update *file*
cat > *file* – places standard input into *file*
more *file* – output the contents of *file*
head *file* – output the first 10 lines of *file*
tail *file* – output the last 10 lines of *file*
tail -f *file* – output the contents of *file* as it grows, starting with the last 10 lines

Process Management

ps – display your currently active processes
top – display all running processes
kill *pid* – kill process id *pid*
killall *proc* – kill all processes named *proc* *
bg – lists stopped or background jobs; resume a stopped job in the background
fg – brings the most recent job to foreground
fg *n* – brings job *n* to the foreground

File Permissions

chmod *octal file* – change the permissions of *file* to *octal*, which can be found separately for user, group, and world by adding:

- 4 – read (r)
- 2 – write (w)
- 1 – execute (x)

Examples:
chmod 777 – read, write, execute for all
chmod 755 – rwx for owner, rx for group and world
For more options, see **man chmod**.

SSH

ssh *user@host* – connect to *host* as *user*
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 keyed or passwordless login

Searching

grep *pattern files* – search for *pattern* in *files*
grep -r *pattern dir* – search recursively for *pattern* in *dir*
command* | grep *pattern – search for *pattern* in the output of *command*
locate *file* – find all instances of *file*

System Info

date – show the current date and time
cal – show this month's calendar
uptime – show current uptime
w – display who is online
whoami – who you are logged in as
finger *user* – display information about *user*
uname -a – show kernel information
cat /proc/cpuinfo – cpu information
cat /proc/meminfo – memory information
man *command* – show the manual for *command*
df – show disk usage
du – show directory space usage
free – show memory and swap usage
whereis *app* – show possible locations of *app*
which *app* – show which *app* will be run by default

Compression

tar cf *file.tar files* – create a tar named *file.tar* containing *files*
tar xf *file.tar* – extract the files from *file.tar*
tar czf *file.tar.gz files* – create a tar with Gzip compression
tar xzf *file.tar.gz* – extract a tar using Gzip
tar cjf *file.tar.bz2* – create a tar with Bzip2 compression
tar xjf *file.tar.bz2* – extract a tar using Bzip2
gzip *file* – compresses *file* and renames it to *file.gz*
gzip -d *file.gz* – decompresses *file.gz* back to *file*

Network

ping *host* – ping *host* and output results
whois *domain* – get whois information for *domain*
dig *domain* – get DNS information for *domain*
dig -x *host* – reverse lookup *host*
wget *file* – download *file*
wget -c *file* – continue a stopped download

Installation

Install from source:
./configure
make
make install
dpkg -i *pkg.deb* – install a package (Debian)
rpm -Uvh *pkg.rpm* – install a package (RPM)

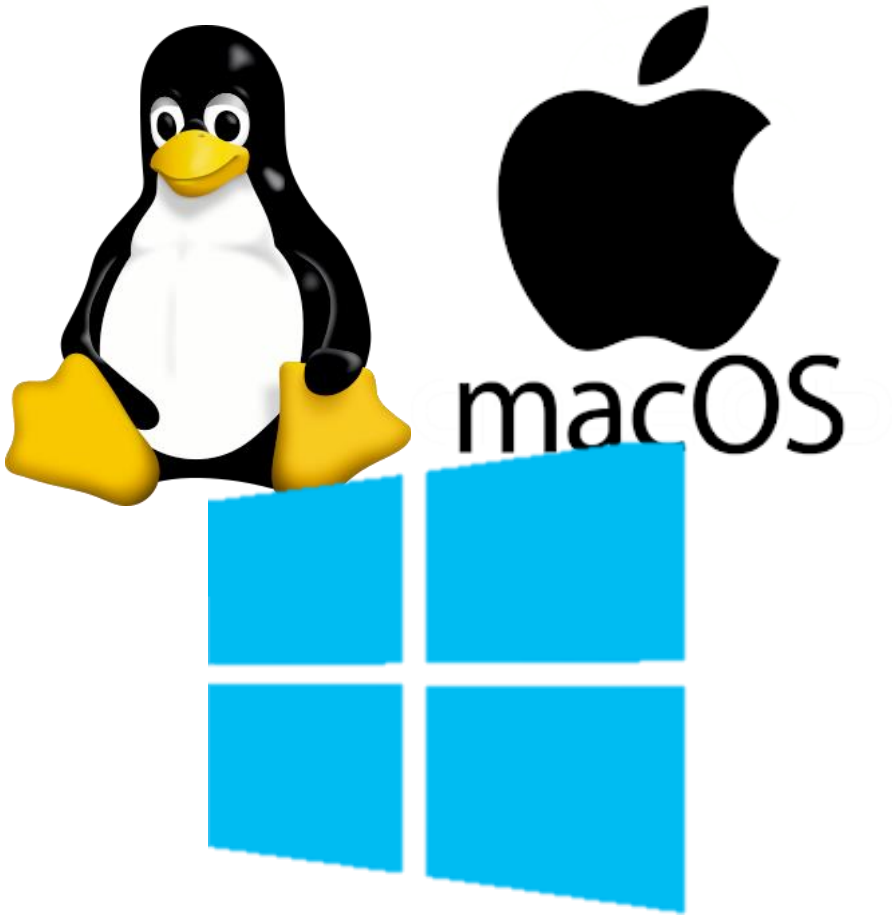
Shortcuts

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+U – erases the whole line
Ctrl+R – type to bring up a recent command
!! - repeats the last command
exit – log out of current session

* use with extreme caution.

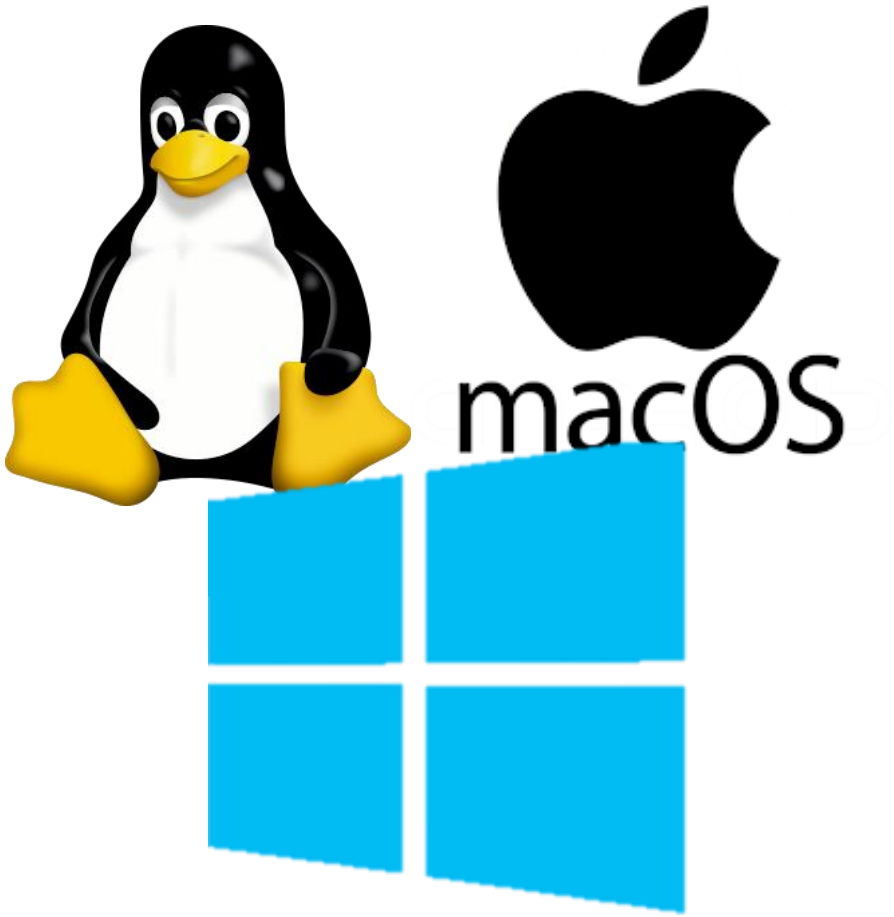
What is UNIX?

Operating system



What is UNIX?

Operating system

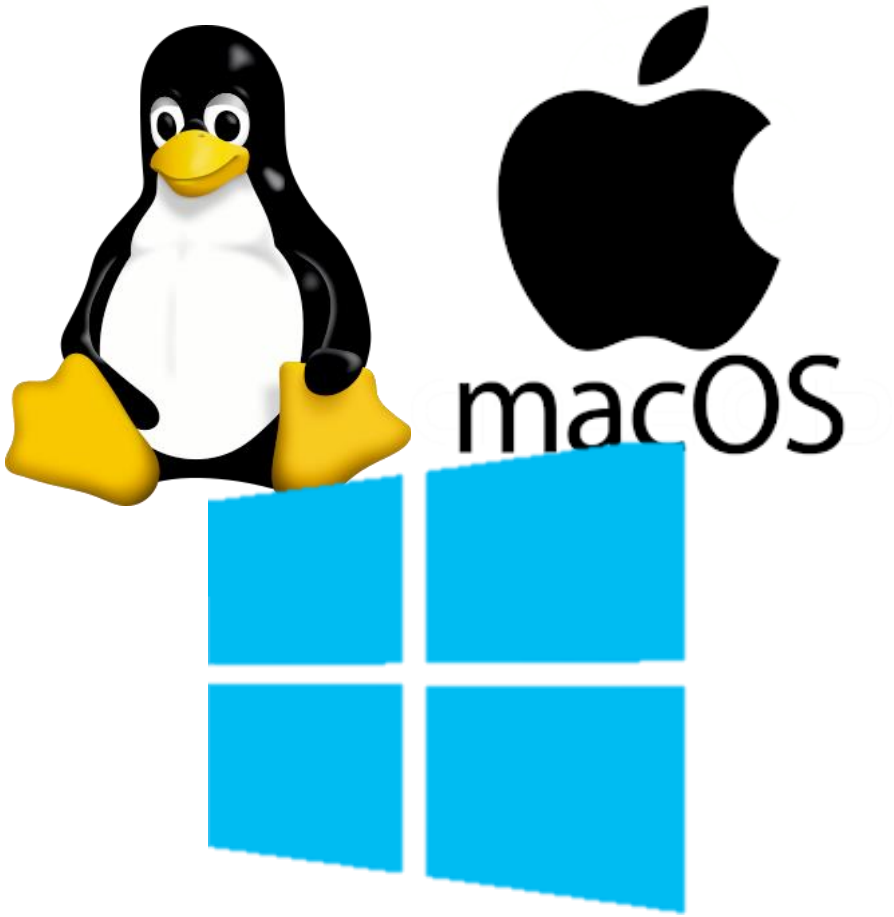


Why do we use it?

- 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?

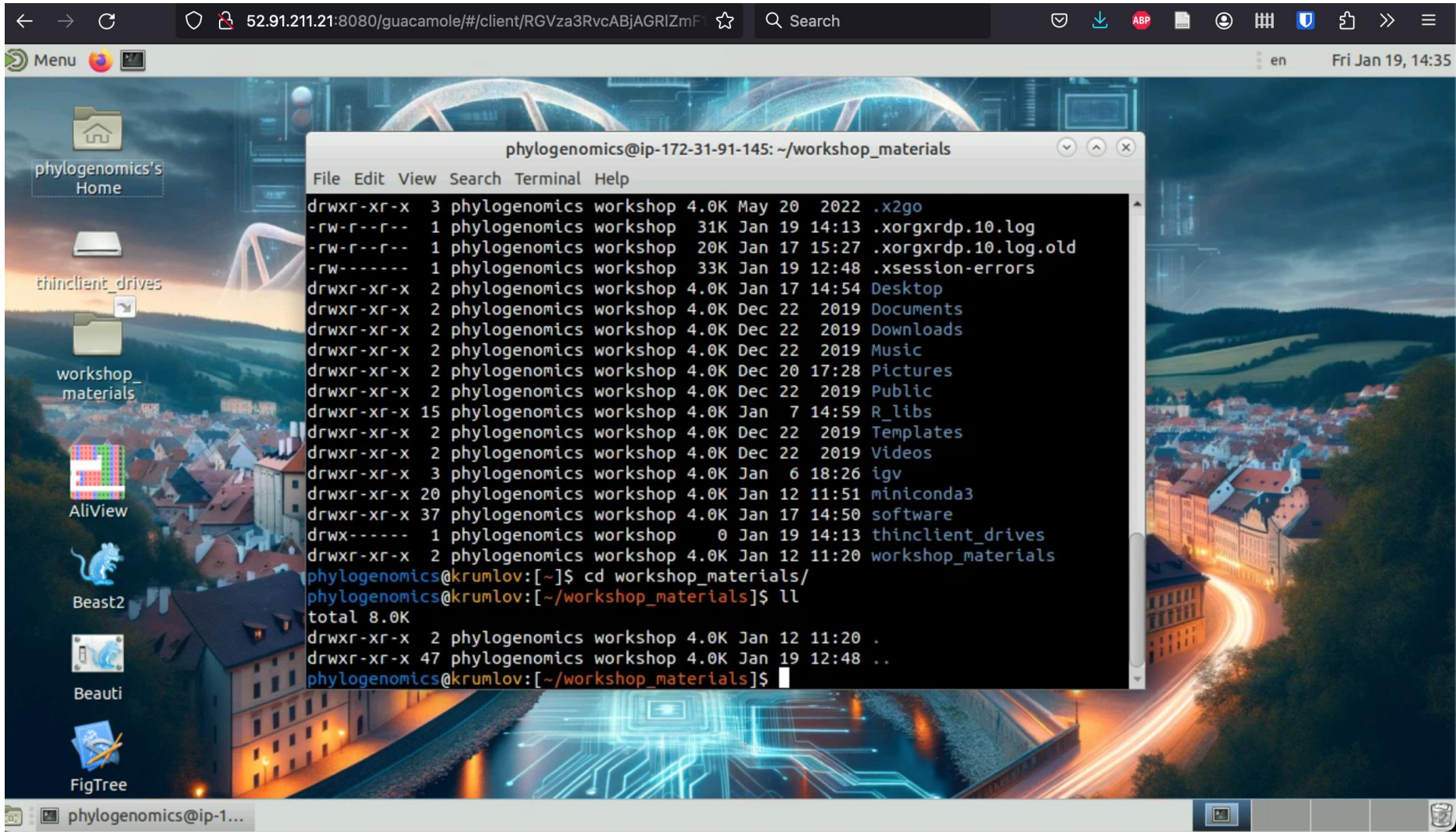
- Linux computers or servers
- Computer clusters
- The cloud

AWS availability zones

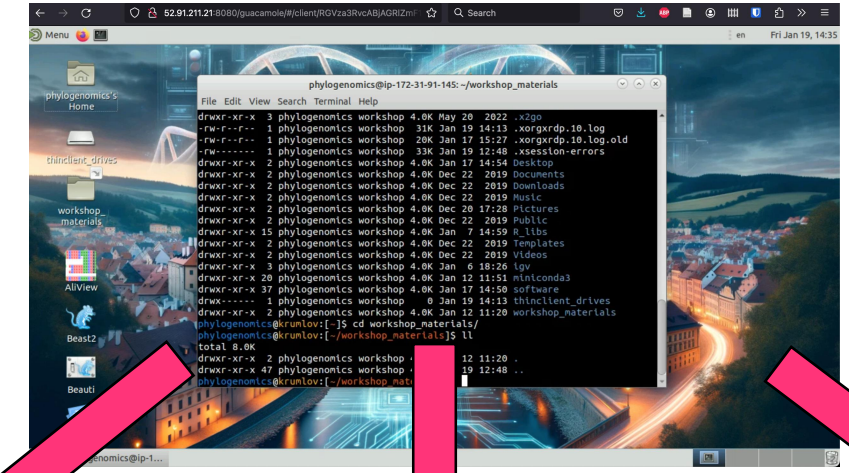
How it works

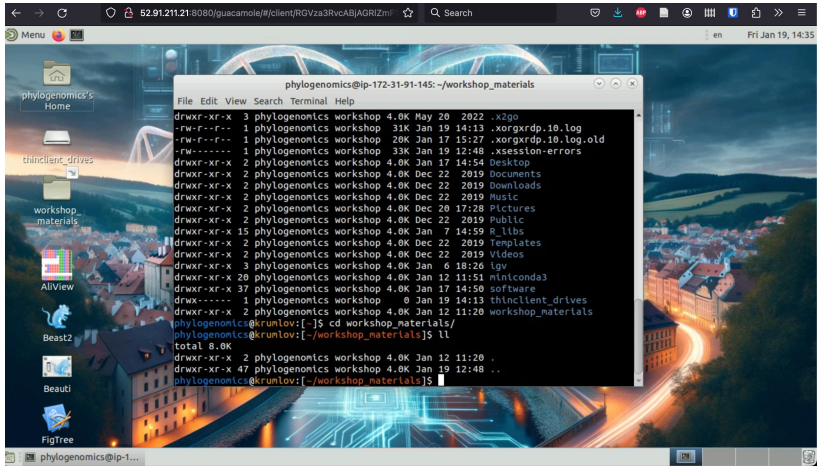
AMI
("Amazon Machine Image")

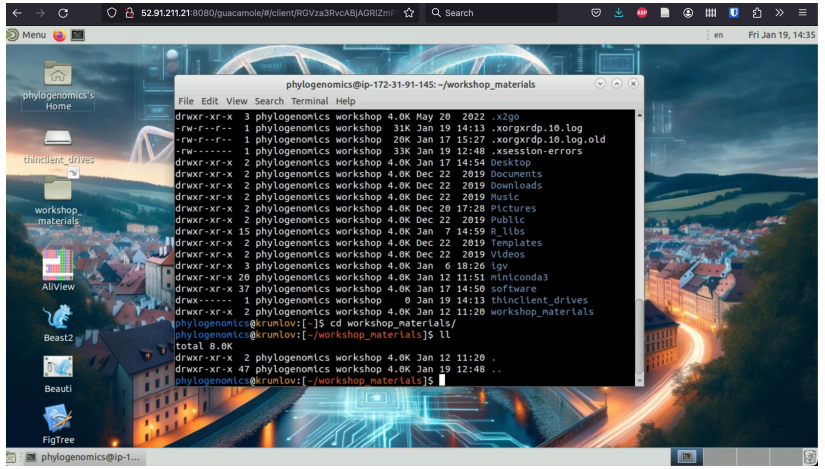
Base computer with all data and software



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.

Watch & listen

Follow along on your computer

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

Find your name and copy your IP address

https://docs.google.com/spreadsheets/d/1-vNj1eYBvvqzKp3c1K

Search

Workshop on Phyloenomics Instance List

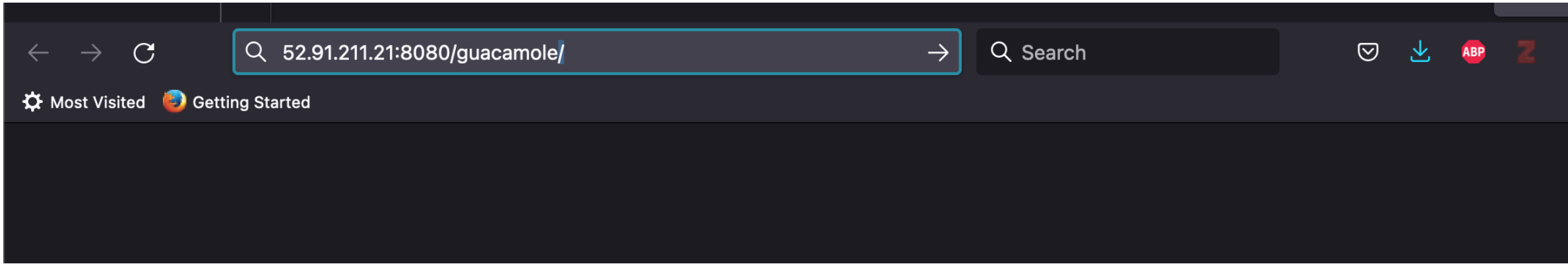
File Edit View Insert Format Data Tools Extensions Help

Share

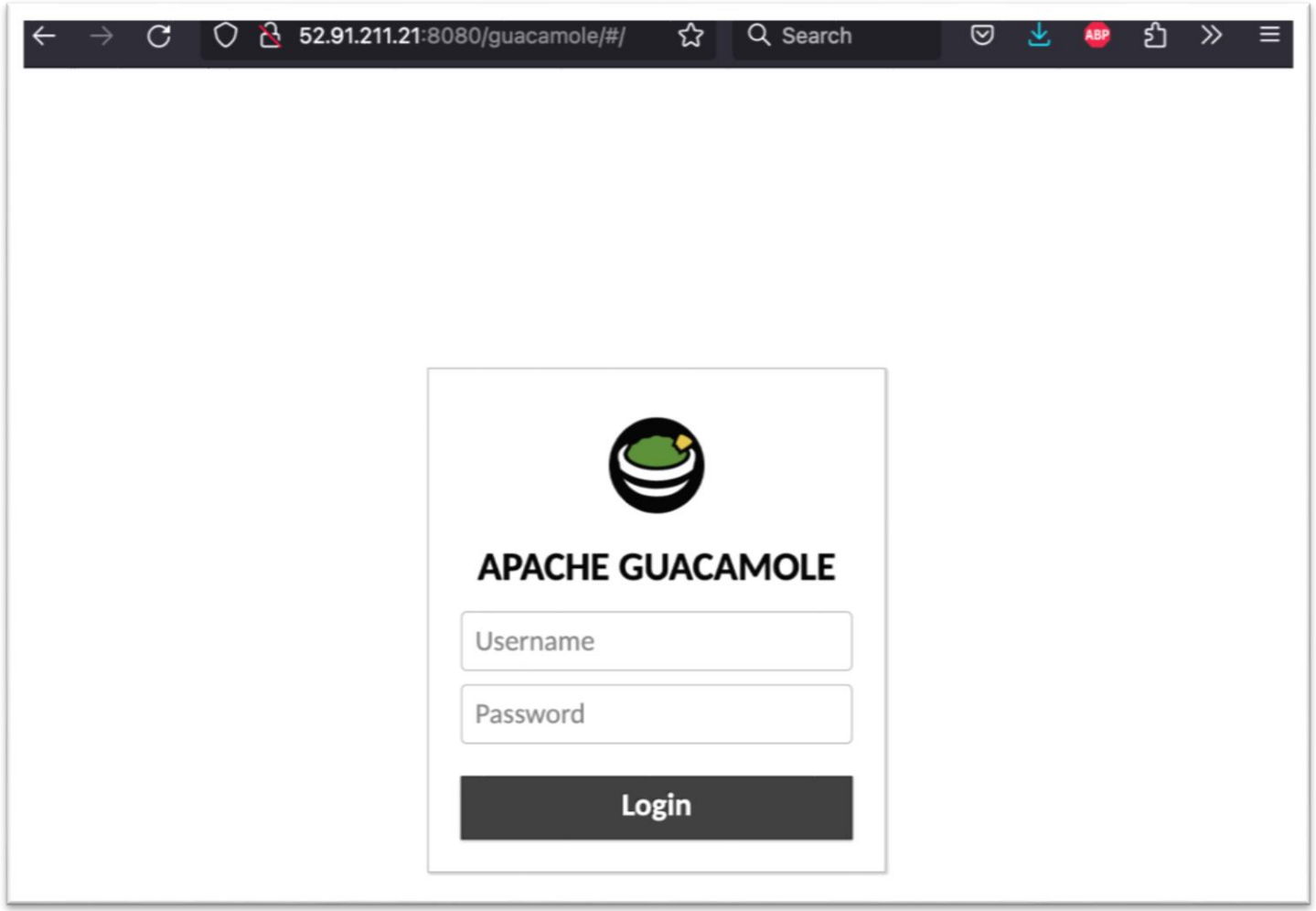
A3 | Phylo-Karin

	A	B	C	D	E	F
1	**Name**	**IP address**	**Guacamole connection**	**ssh connection**	**RStudio server connection**	**Username: phyloge**
2	guy	3.2.144.5	3.2.144.5:8080/guacamole	ssh genomics@3.2.144.5	3.2.144.5:8787	**Date:**
3	Phylo-Karin	52.91.211.21	52.91.211.21:8080/guacamole	ssh genomics@52.91.211.21	52.91.211.21:8787	
4	Phylo-Marina	44.212.5.42	44.212.5.42:8080/guacamole	ssh genomics@44.212.5.42	44.212.5.42:8787	
5	Phylo-Jacob	3.89.92.242	3.89.92.242:8080/guacamole	ssh genomics@3.89.92.242	3.89.92.242:8787	
6	Phylo-Gemma	54.205.120.236	54.205.120.236:8080/guacamole	ssh genomics@54.205.120.236	54.205.120.236:8787	
7	Phylo-Michal	44.207.6.114	44.207.6.114:8080/guacamole	ssh genomics@44.207.6.114	44.207.6.114:8787	
8	Phylo-Rosa	3.94.82.239	3.94.82.239:8080/guacamole	ssh genomics@3.94.82.239	3.94.82.239:8787	
9			:8080/guacamole	ssh genomics@	:8787	
10			:8080/guacamole	ssh genomics@	:8787	
11			:8080/guacamole	ssh genomics@	:8787	
12			:8080/guacamole	ssh genomics@	:8787	
13			:8080/guacamole	ssh genomics@	:8787	
14			:8080/guacamole	ssh genomics@	:8787	

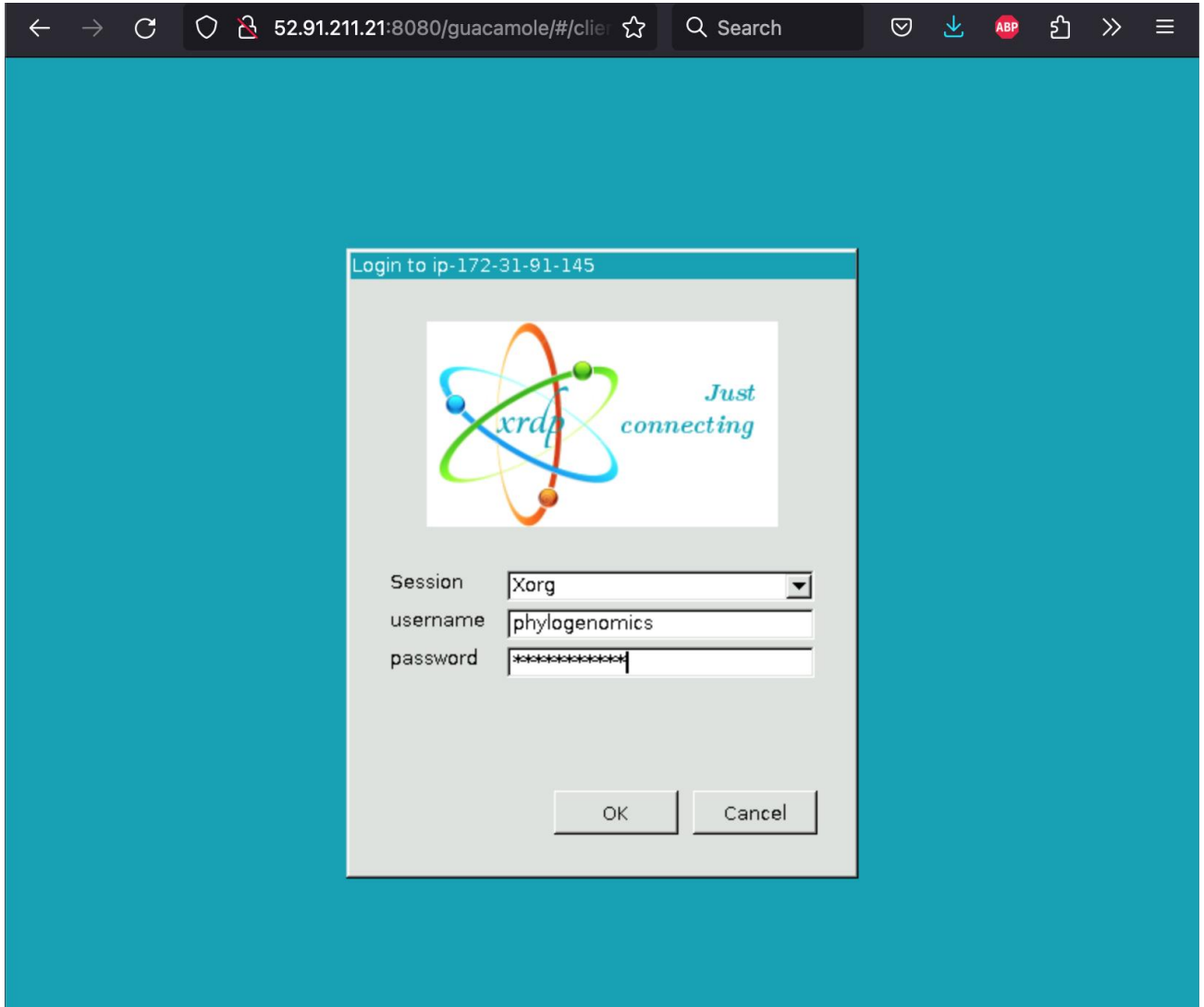
- Open your internet browser (e.g. Google Chrome)
- Paste the IP address followed by ‘:8080/guacamole’
- 52.91.211.21:8080/guacamole



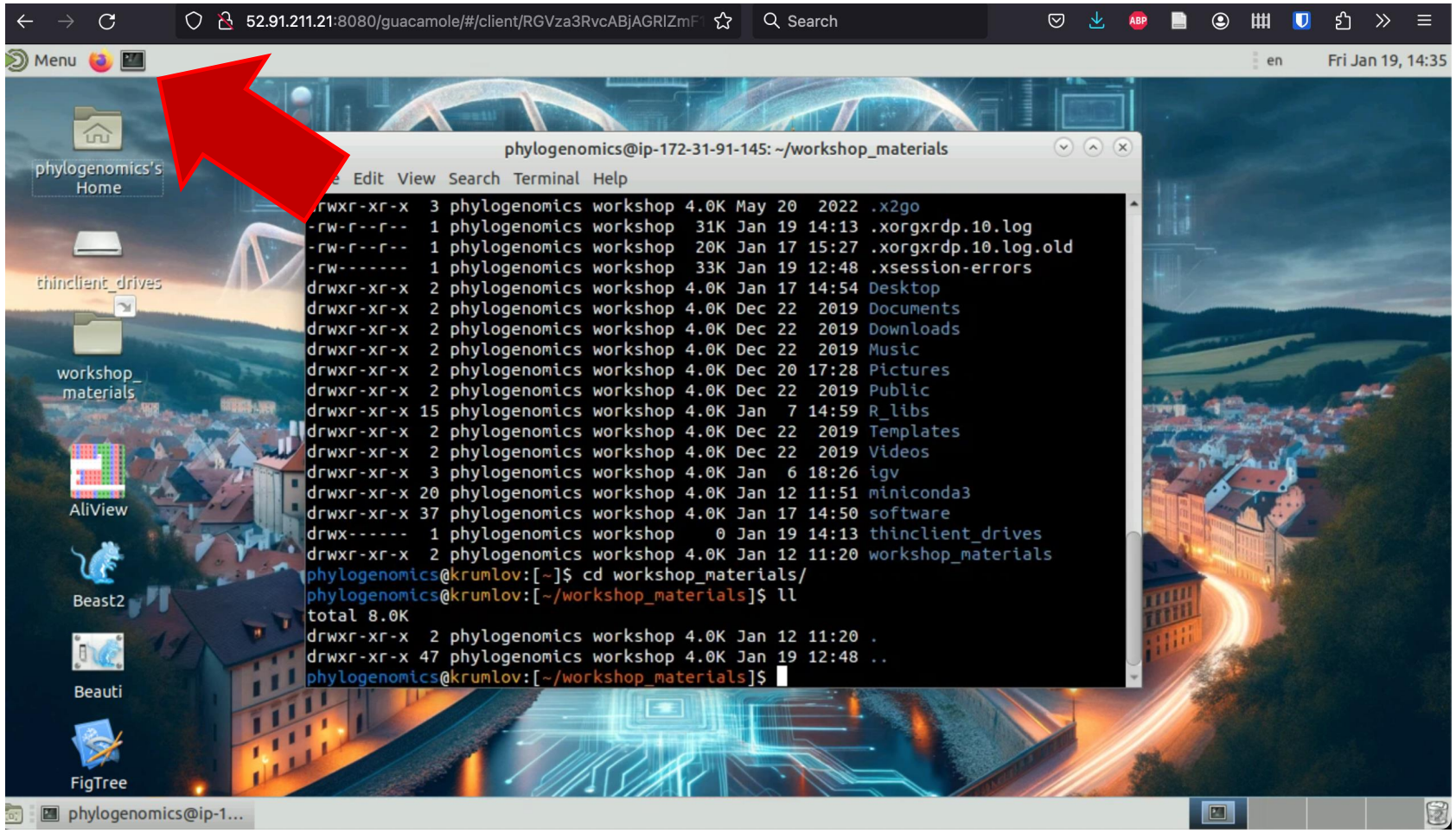
Enter the username “phylogenomics” and password



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

https://docs.google.com/spreadsheets/d/1-vNj1eYBvvqzKp3c1K

Search

Workshop on Phyloenomics Instance List

File Edit View Insert Format Data Tools Extensions Help

Share

A3 | Phylo-Karin

	A	B	C	D	E	F
1	**Name**	**IP address**	**Guacamole connection**	**ssh connection**	**RStudio server connection**	**Username: phyloge**
2	guy	3.2.144.5	3.2.144.5:8080/guacamole	ssh genomics@3.2.144.5	3.2.144.5:8787	**Date:**
3	Phylo-Karin	52.91.211.21	52.91.211.21:8080/guacamole	ssh genomics@52.91.211.21	52.91.211.21:8787	
4	Phylo-Marina	44.212.5.42	44.212.5.42:8080/guacamole	ssh genomics@44.212.5.42	44.212.5.42:8787	
5	Phylo-Jacob	3.89.92.242	3.89.92.242:8080/guacamole	ssh genomics@3.89.92.242	3.89.92.242:8787	
6	Phylo-Gemma	54.205.120.236	54.205.120.236:8080/guacamole	ssh genomics@54.205.120.236	54.205.120.236:8787	
7	Phylo-Michal	44.207.6.114	44.207.6.114:8080/guacamole	ssh genomics@44.207.6.114	44.207.6.114:8787	
8	Phylo-Rosa	3.94.82.239	3.94.82.239:8080/guacamole	ssh genomics@3.94.82.239	3.94.82.239:8787	
9			:8080/guacamole	ssh genomics@	:8787	
10			:8080/guacamole	ssh genomics@	:8787	
11			:8080/guacamole	ssh genomics@	:8787	

Copy & Paste

AVOID COPYING AND PASTING WHEREVER POSSIBLE!

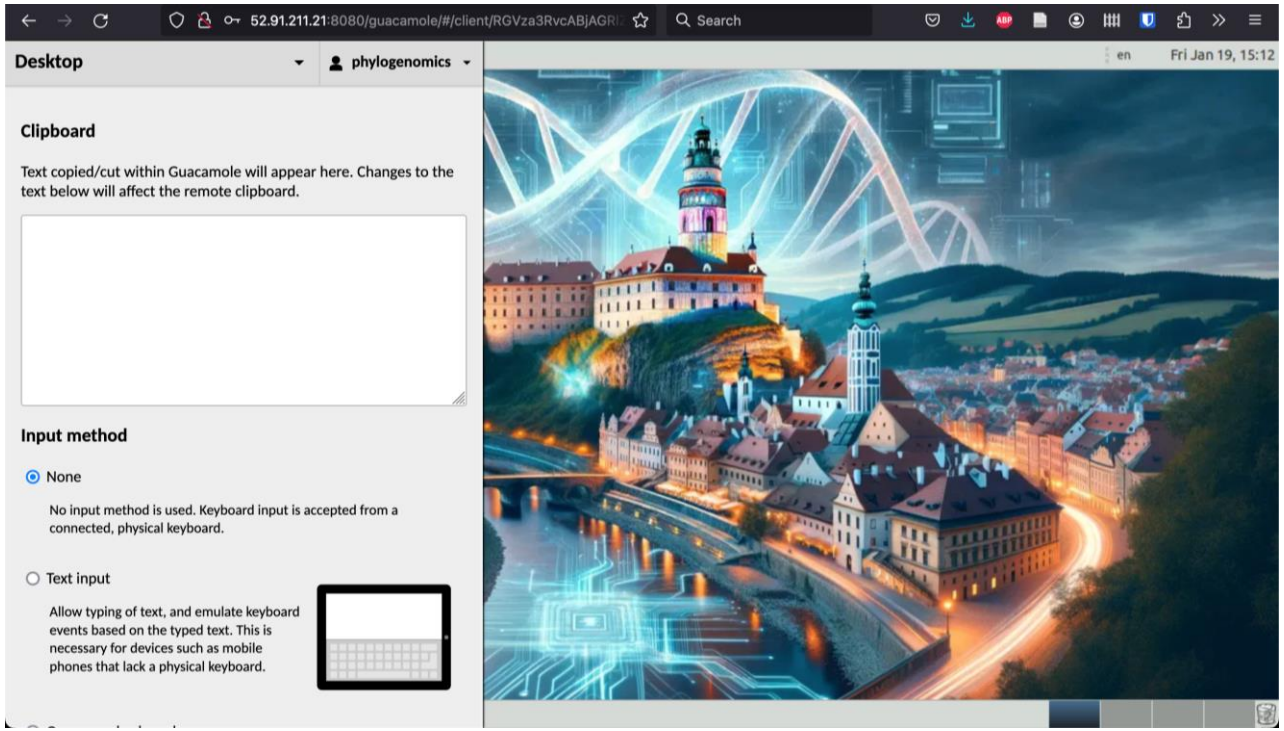
But if you do need to…

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?

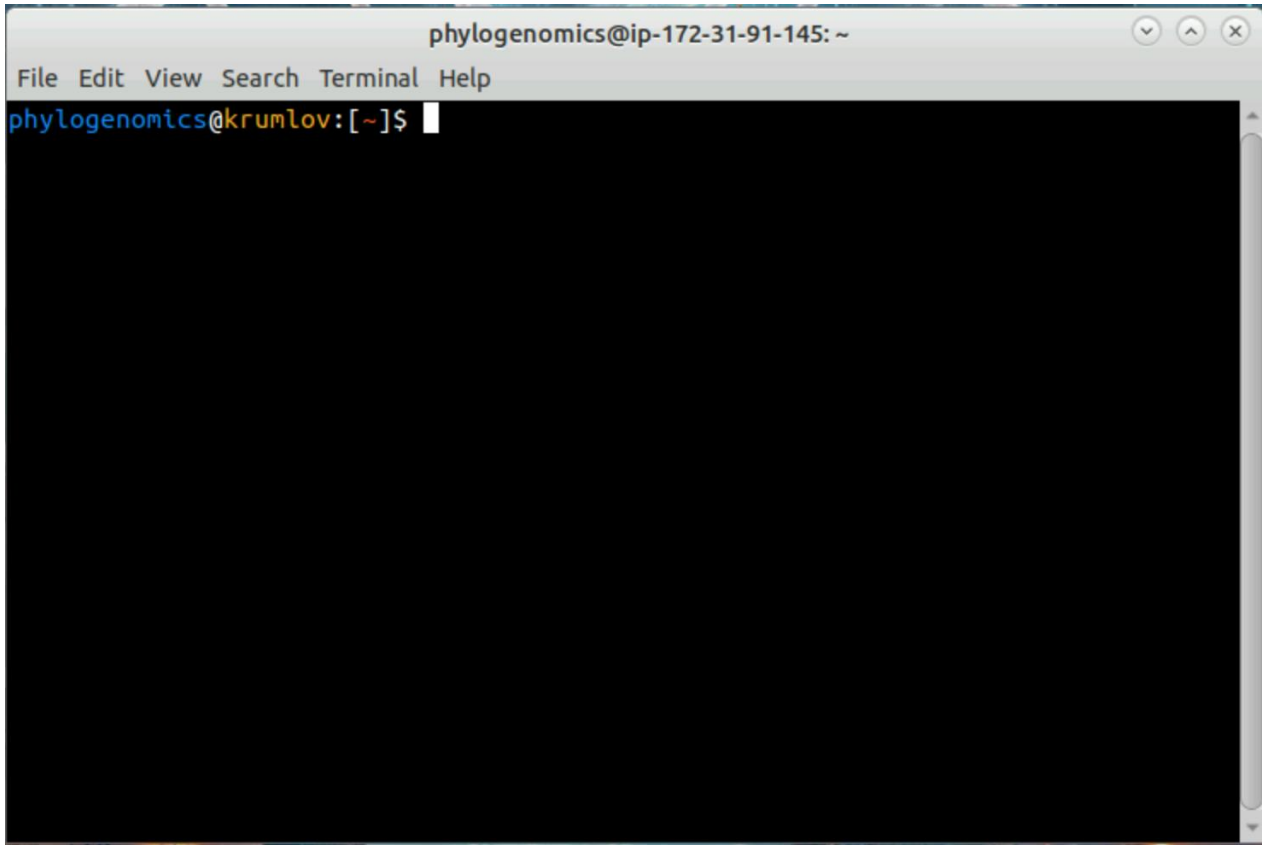
Commands overview

`pwd`	*print working directory*
`~`	*home*
`.`	*here*
`..`	*one directory up*
`mkdir`	*make new directory*
`cd`	*change directory*
`touch`	*create file*
`ls`	*list*
`man`	*manual*
`mv`	*move*
`rm`	*remove*

`cp`	*copy*
`gunzip`	*unzip*
`tar`	*unarchive*
`head`	*first (n=10) lines*
`tail`	*last (n=10) lines*
`cat`	*concatenate*
`wc`	*word count*
`grep`	*pattern search*
`|`	*pipeline*
`sed`	*stream editor*
`chmod`	*change file modes*

The terminal (command line, shell, prompt)

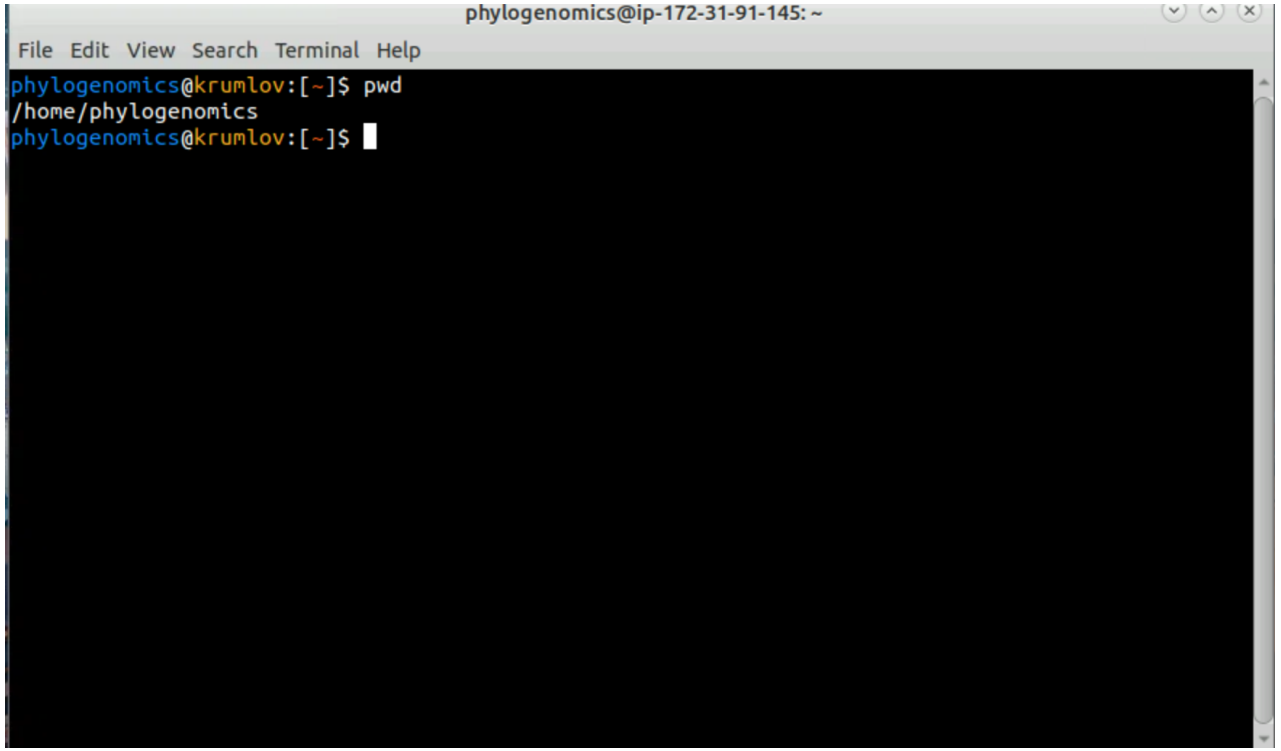
Where you see this “$” followed by text, I want you to type the text on your command line



Location is important

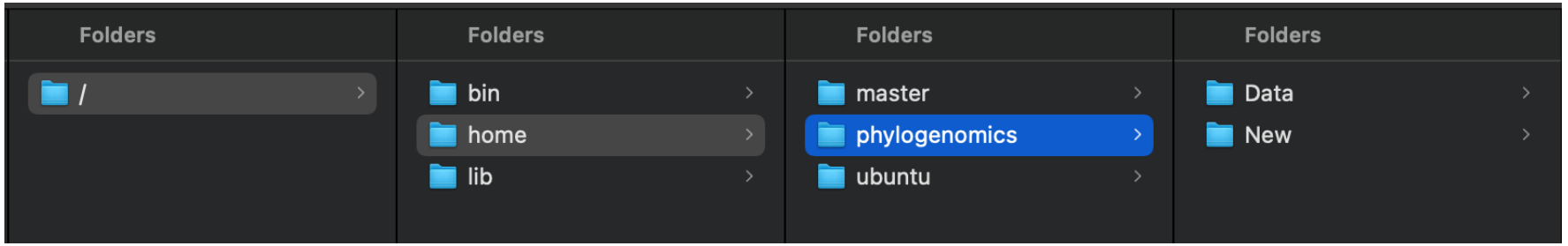
First task: Where am I?

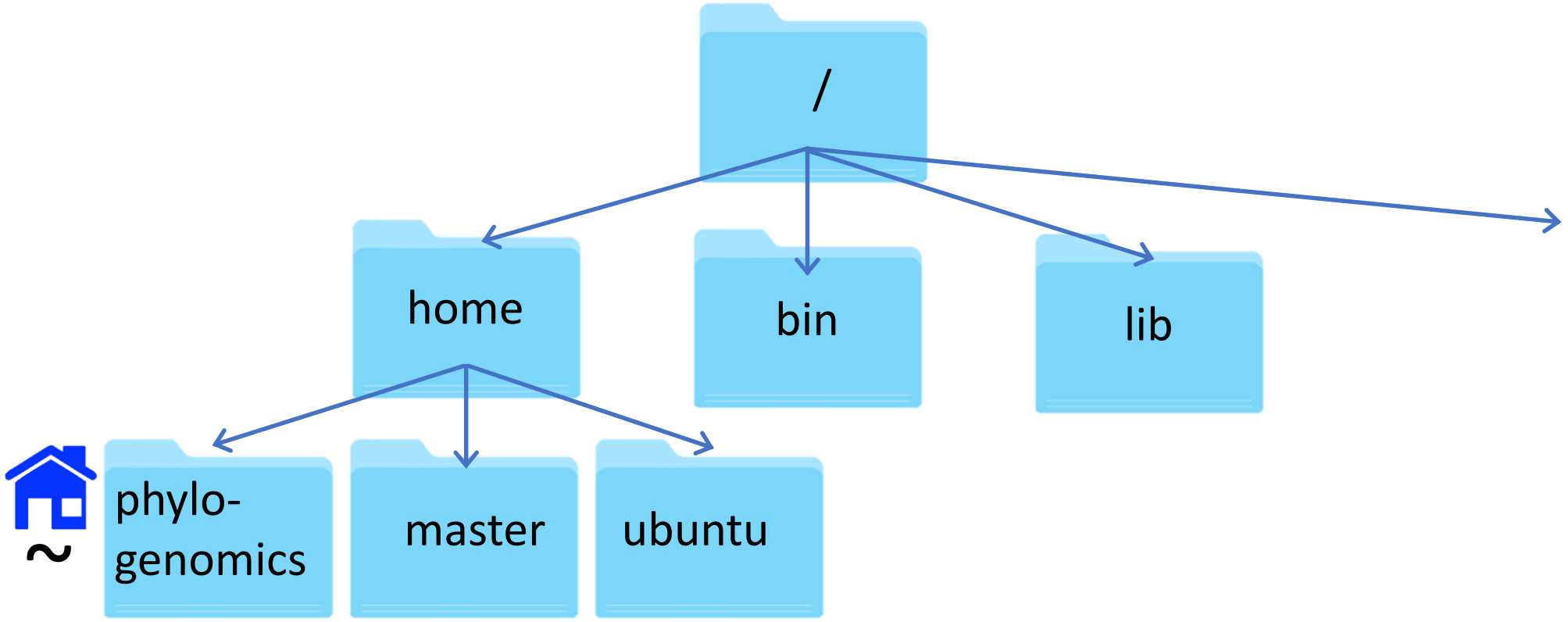
```
$ pwd
```



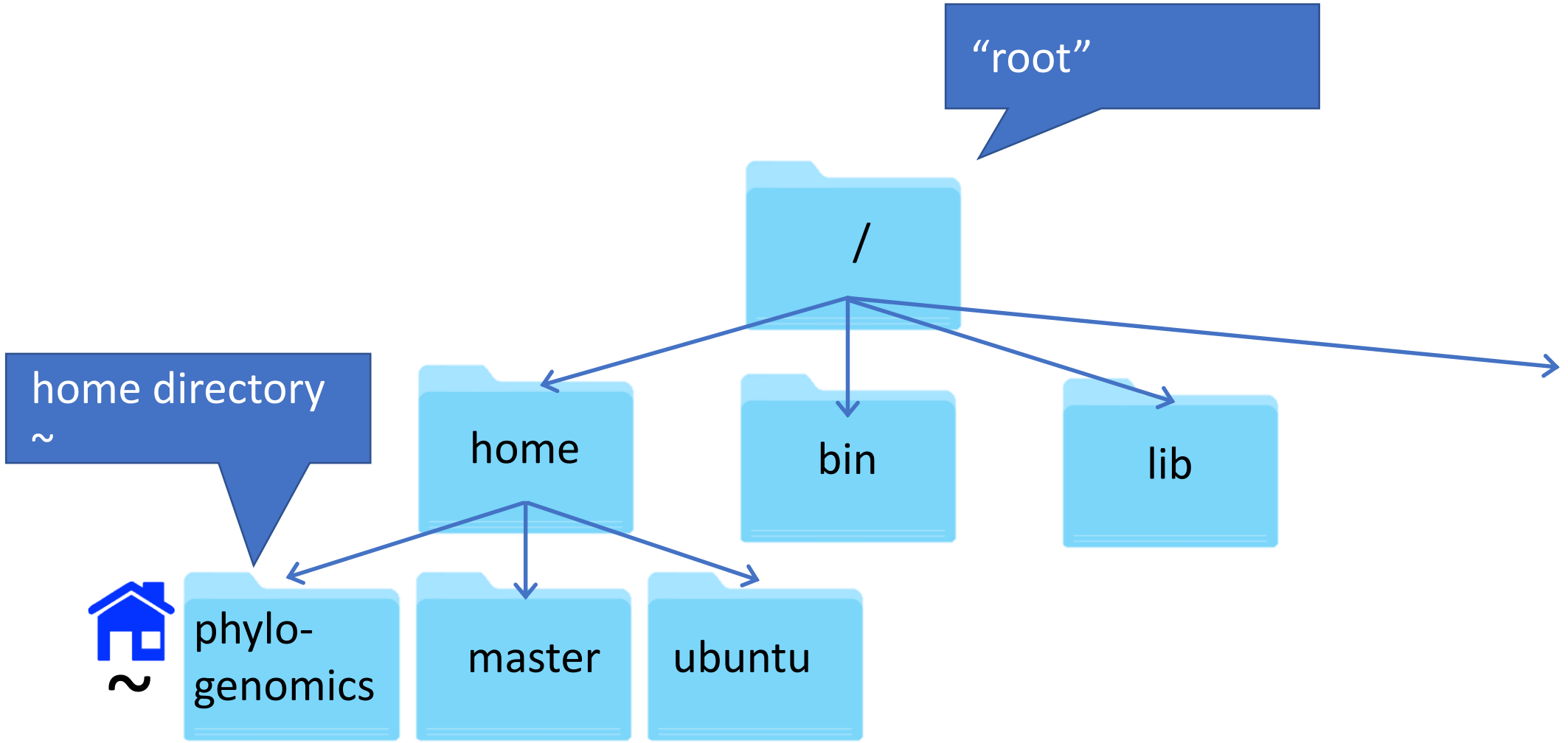
This is your working directory, i.e. where you currently are.

Creating and navigating directories

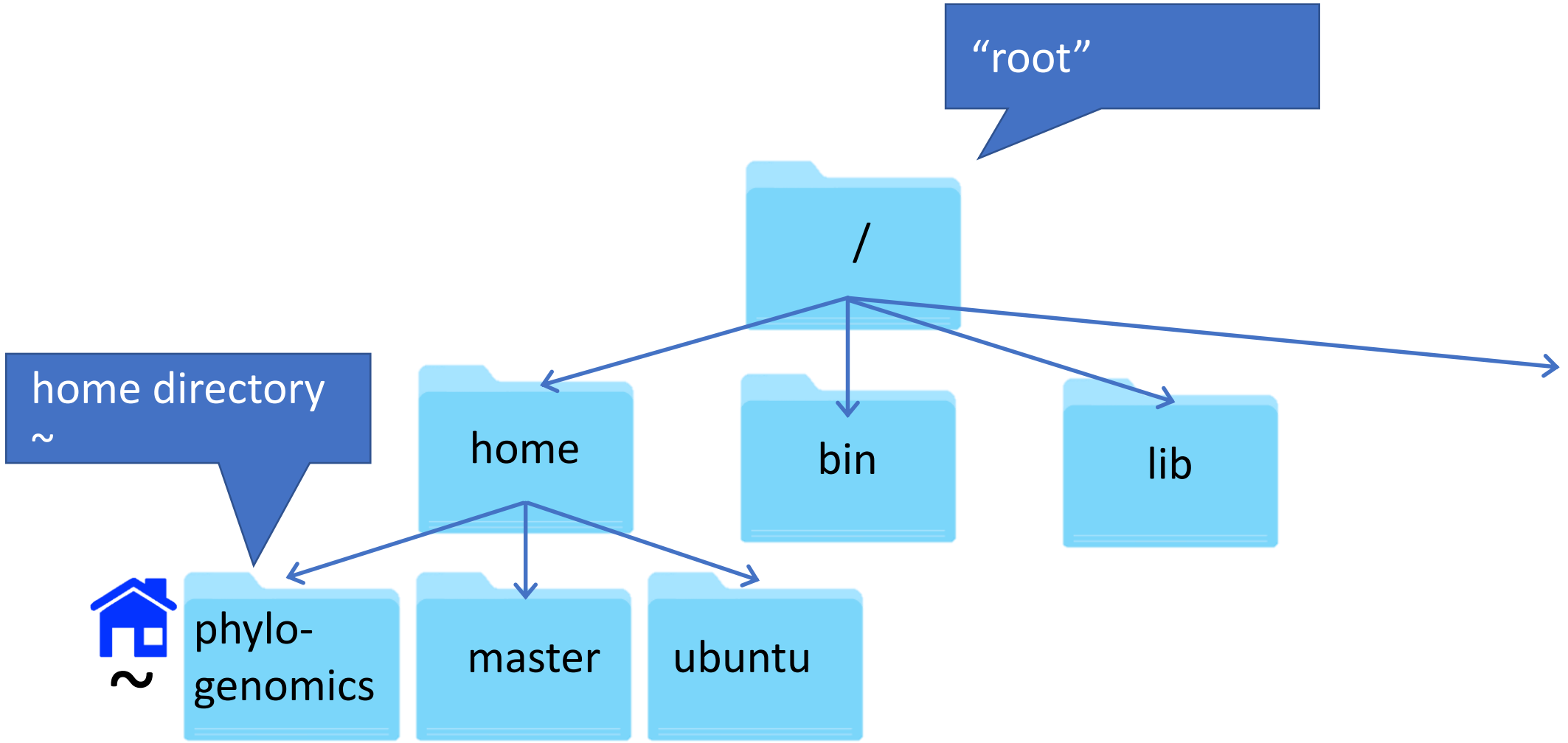




Creating and navigating directories



Creating and navigating directories



Creating and navigating directories

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

```
$ mkdir ./Data
```

Change into the new directory

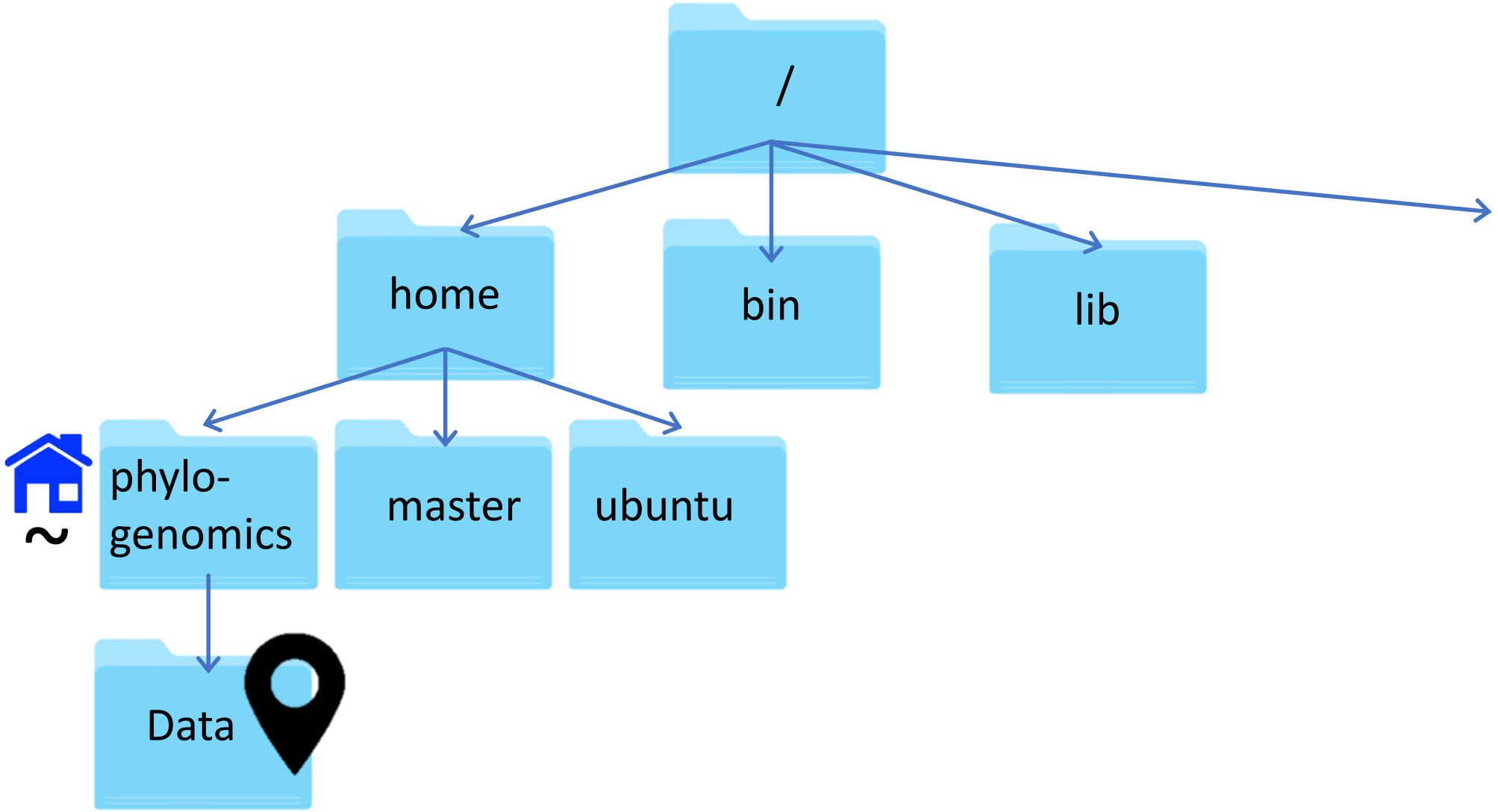
```
$ cd Data
```

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



Creating and navigating directories

Make an empty file ”rags”

```
$ touch rags
```

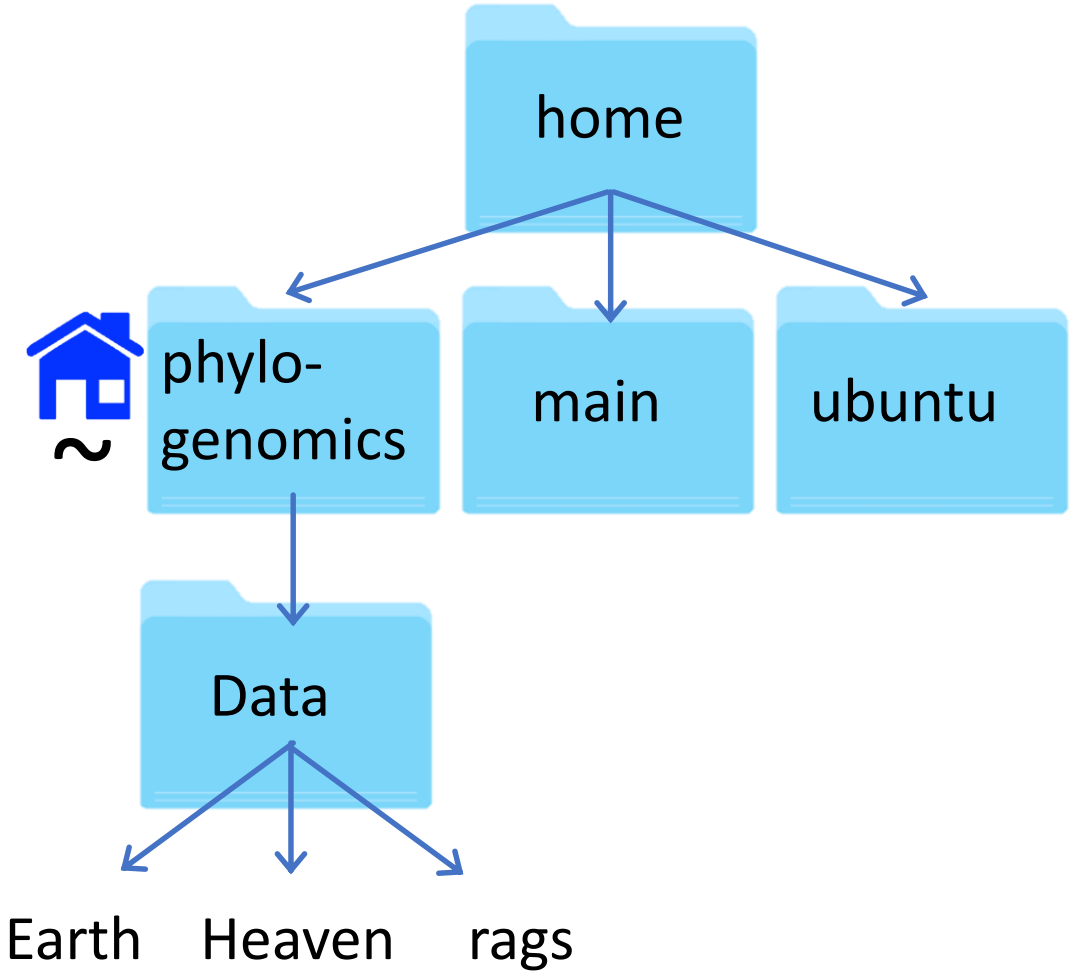
And another two “Heaven” and “Earth”

```
$ touch Heaven Earth
```

Now let’s list the contents of the current directory (Data)

```
$ ls
```

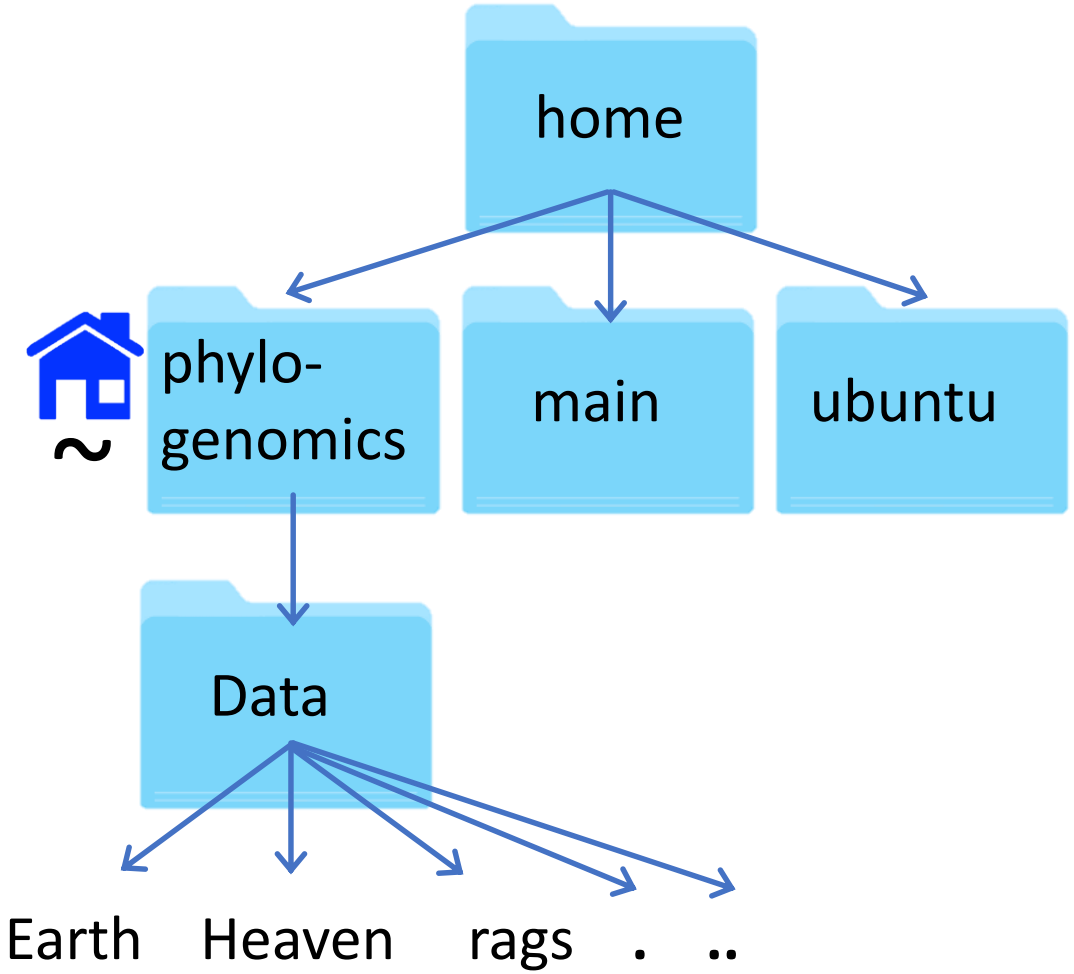
Creating and navigating directories



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

```
$ ls -a
```

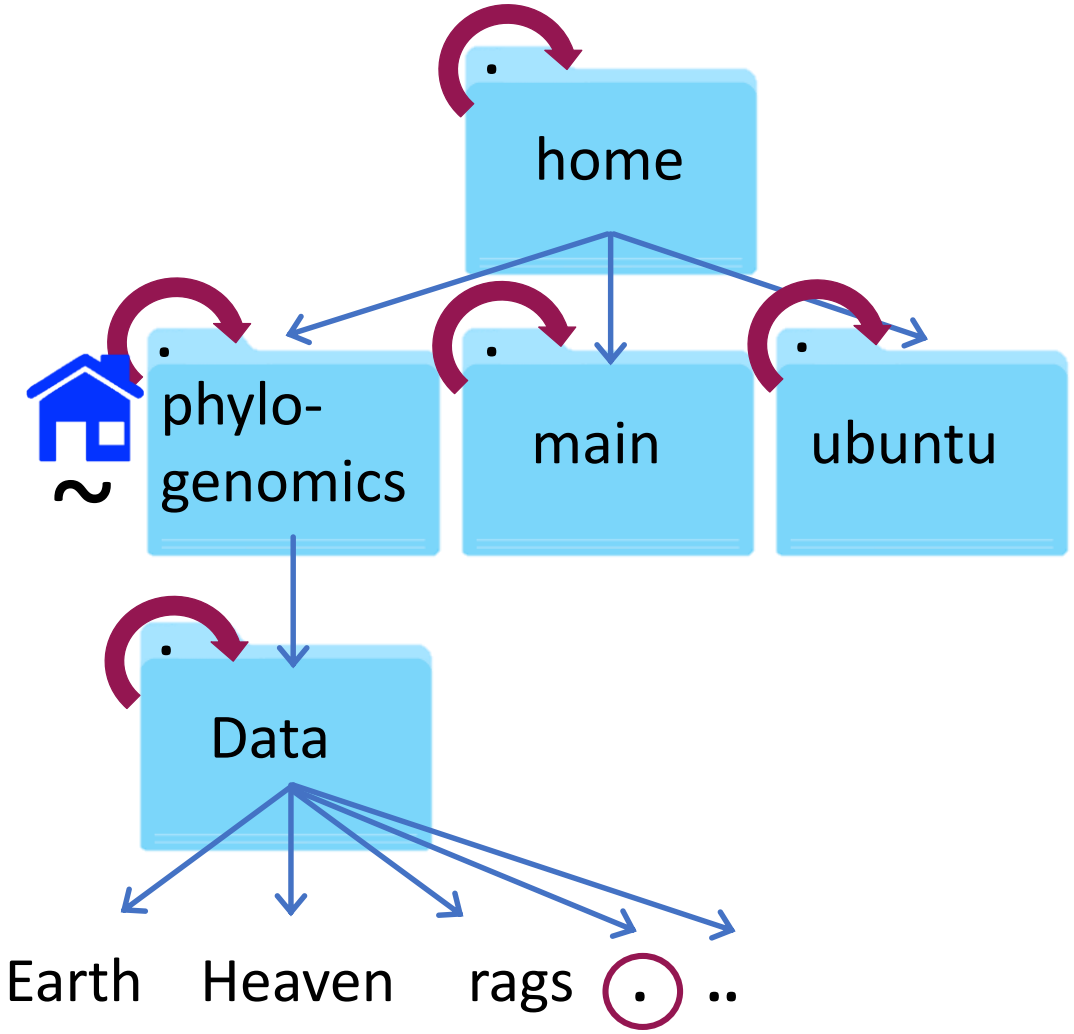
Creating and navigating directories



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

```
$ ls -a
```

Creating and navigating directories



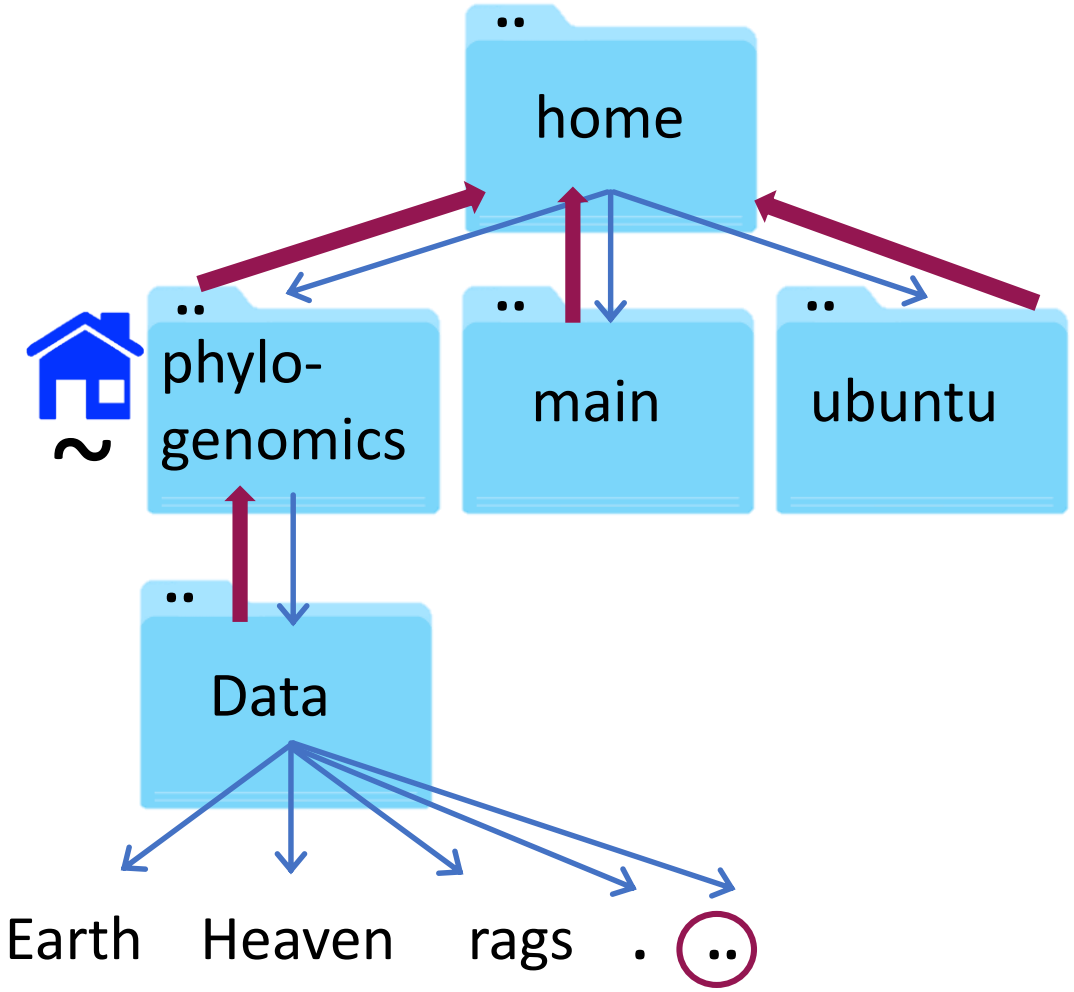
Now list **all** of the files in the directory

```
$ 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

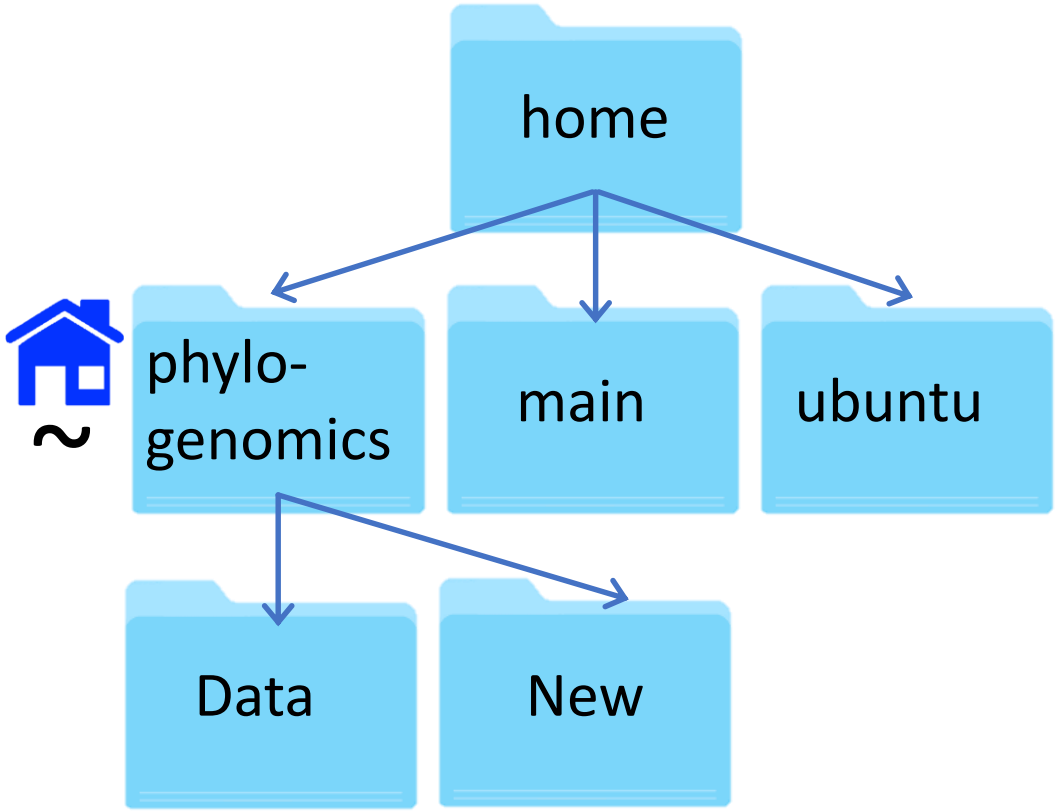
Creating and navigating directories

. 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

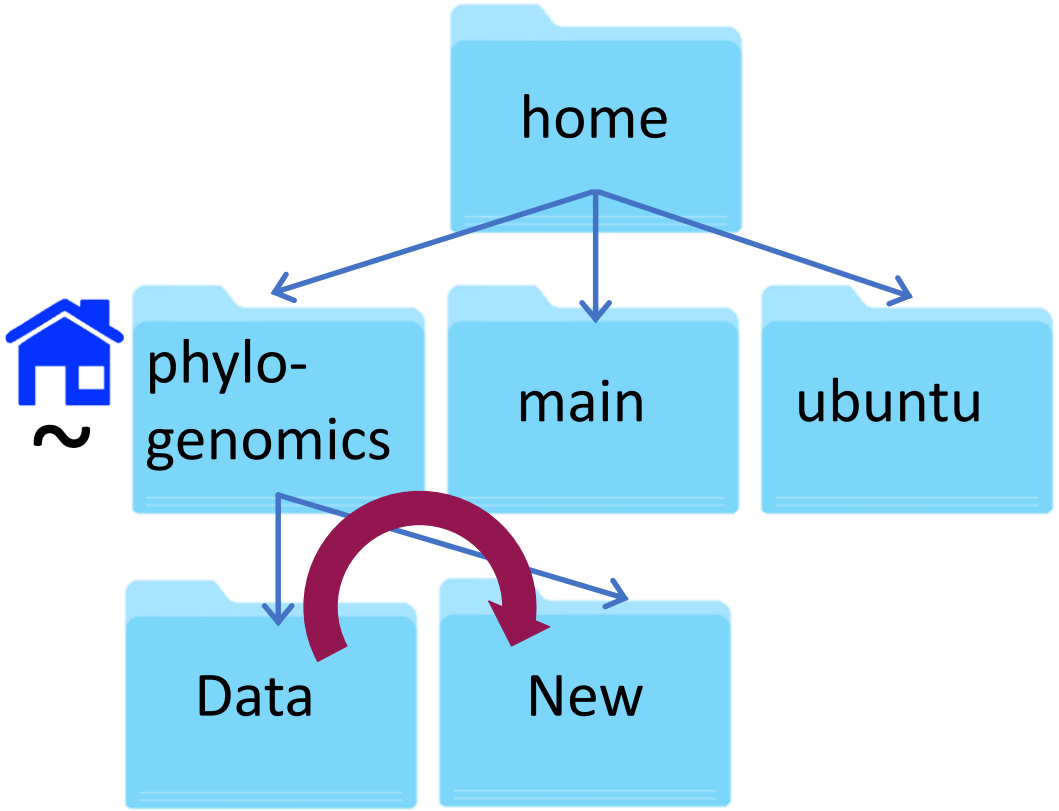
Creating and navigating directories



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

```
$ mkdir ../New
```

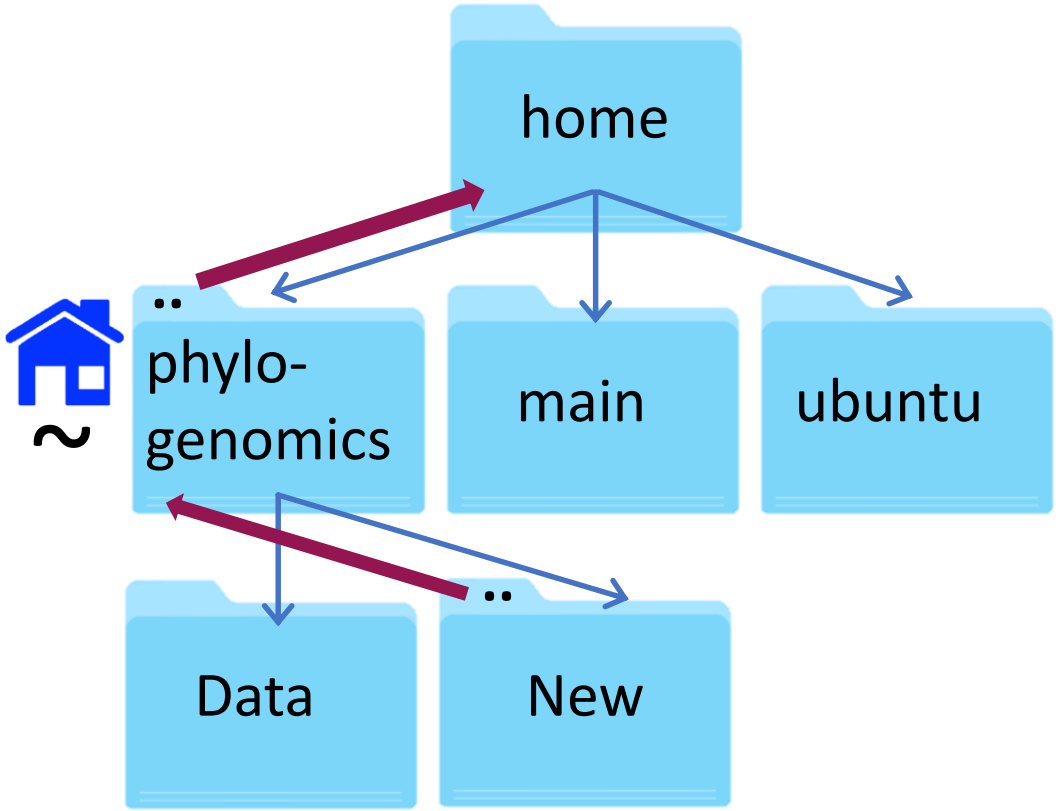
Creating and navigating directories



Move from Data to New
RELATIVE PATH

```
$ cd ../New
```

Creating and navigating directories



Move from New to home
RELATIVE PATH

```
$ cd ../../
```

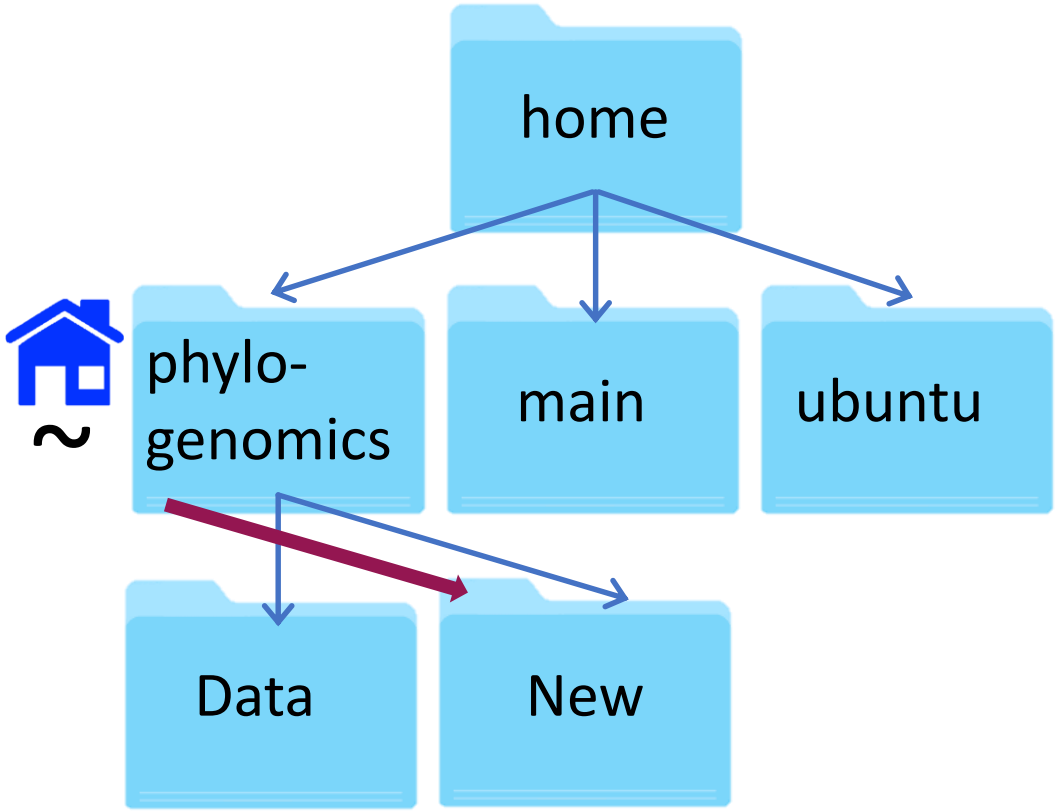
Creating and navigating directories

Relative paths will always change depending on your location.

The alternative is ABSOLUTE paths.

These always start from root and will never change.

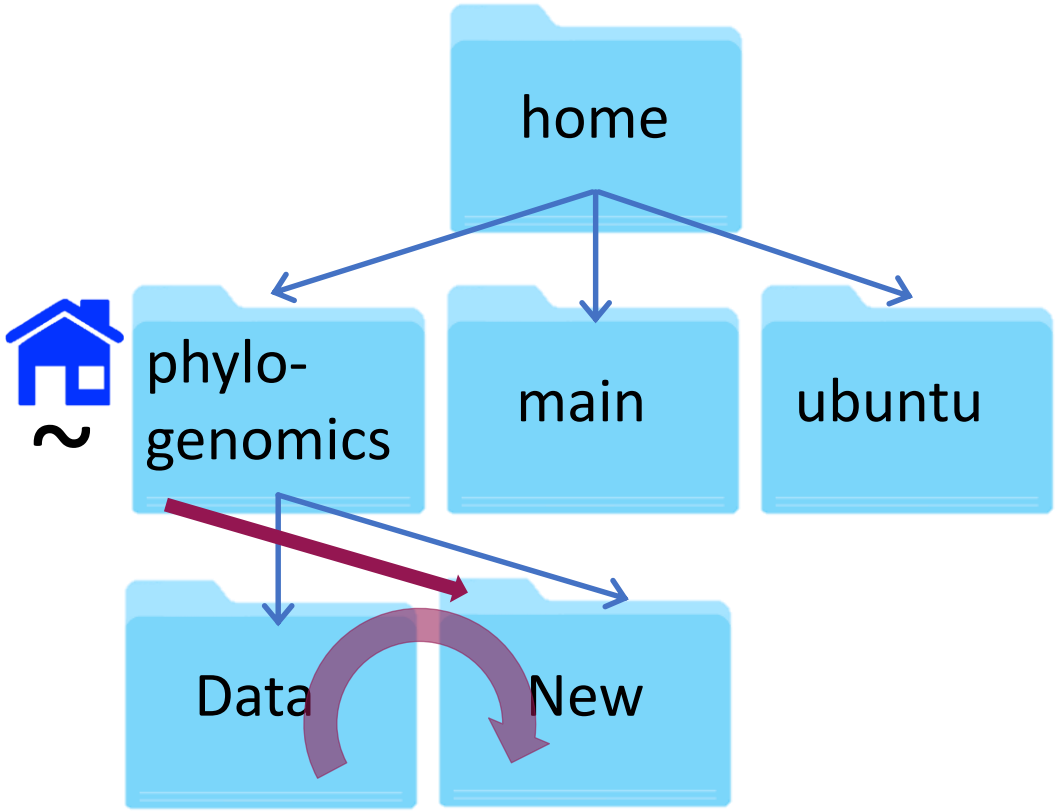
Creating and navigating directories



Move from your home directory to New
ABSOLUTE PATH

```
$ 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`

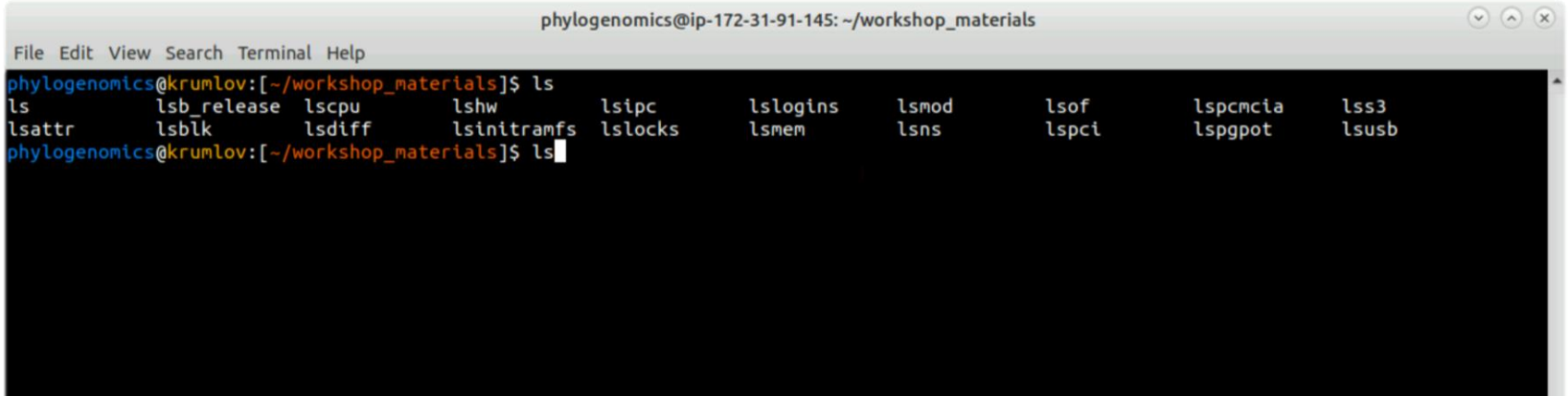
CHALLENGE 1!

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!

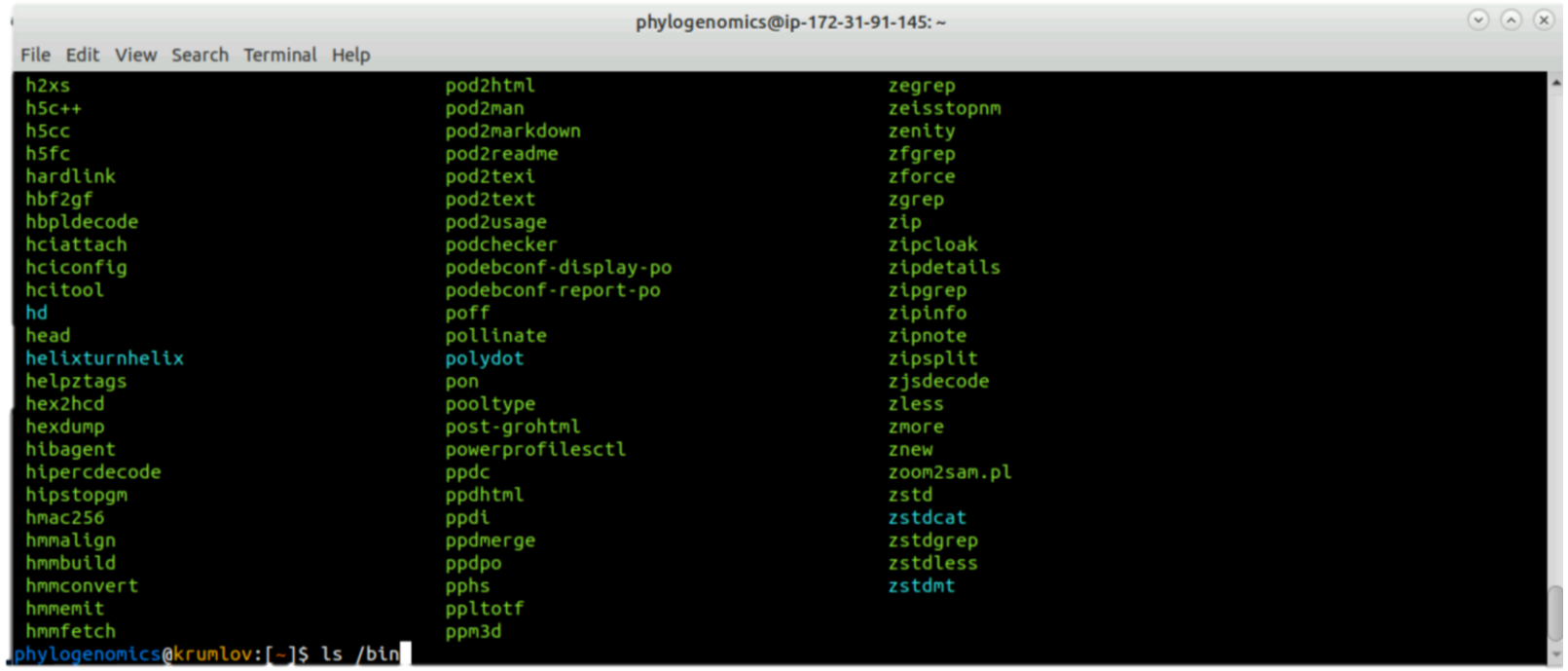
If you want all of your previous commands listed, simply type `$ history`

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) and 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
```

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

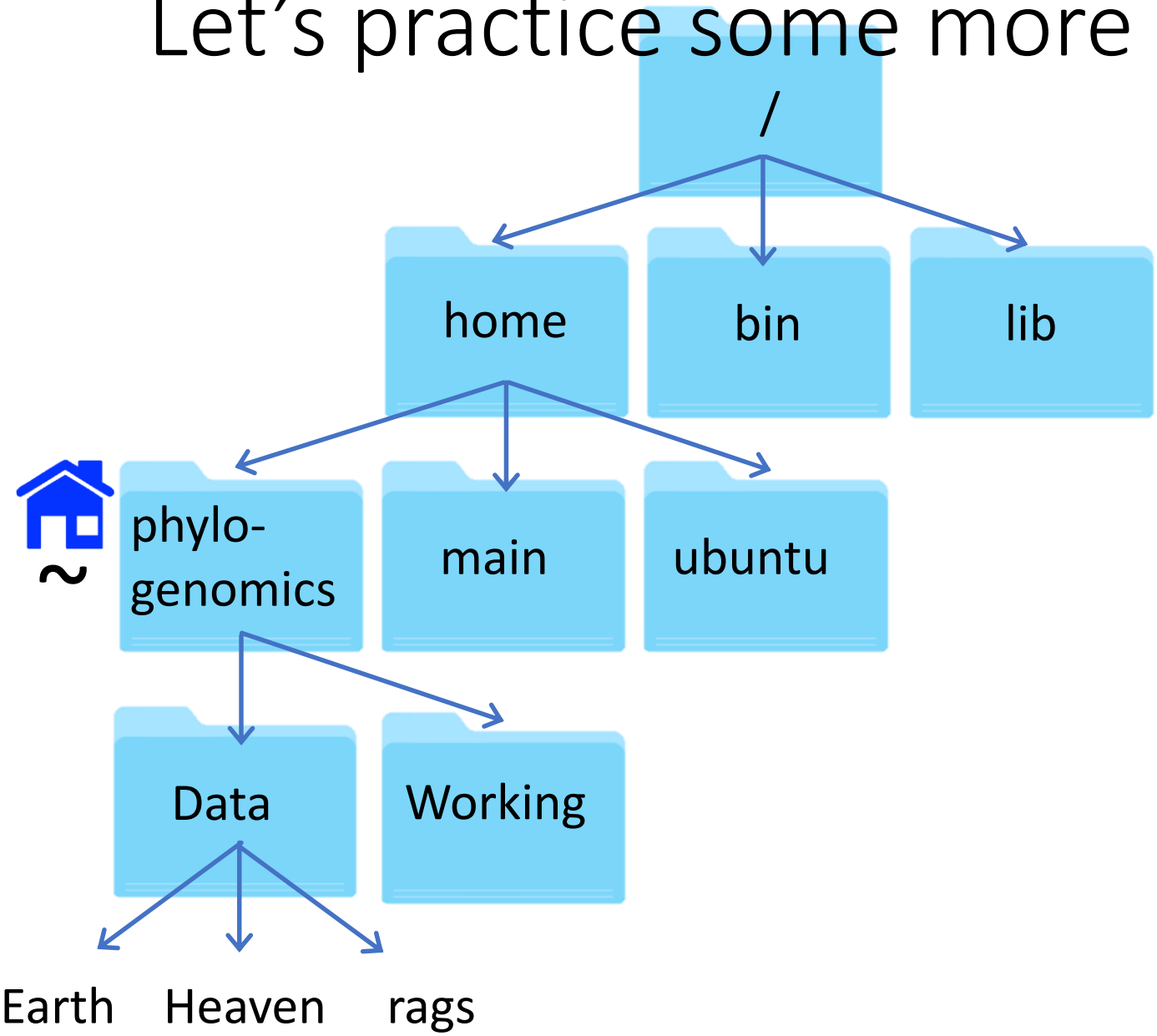
```
$ /usr/bin/mkdir
```

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.

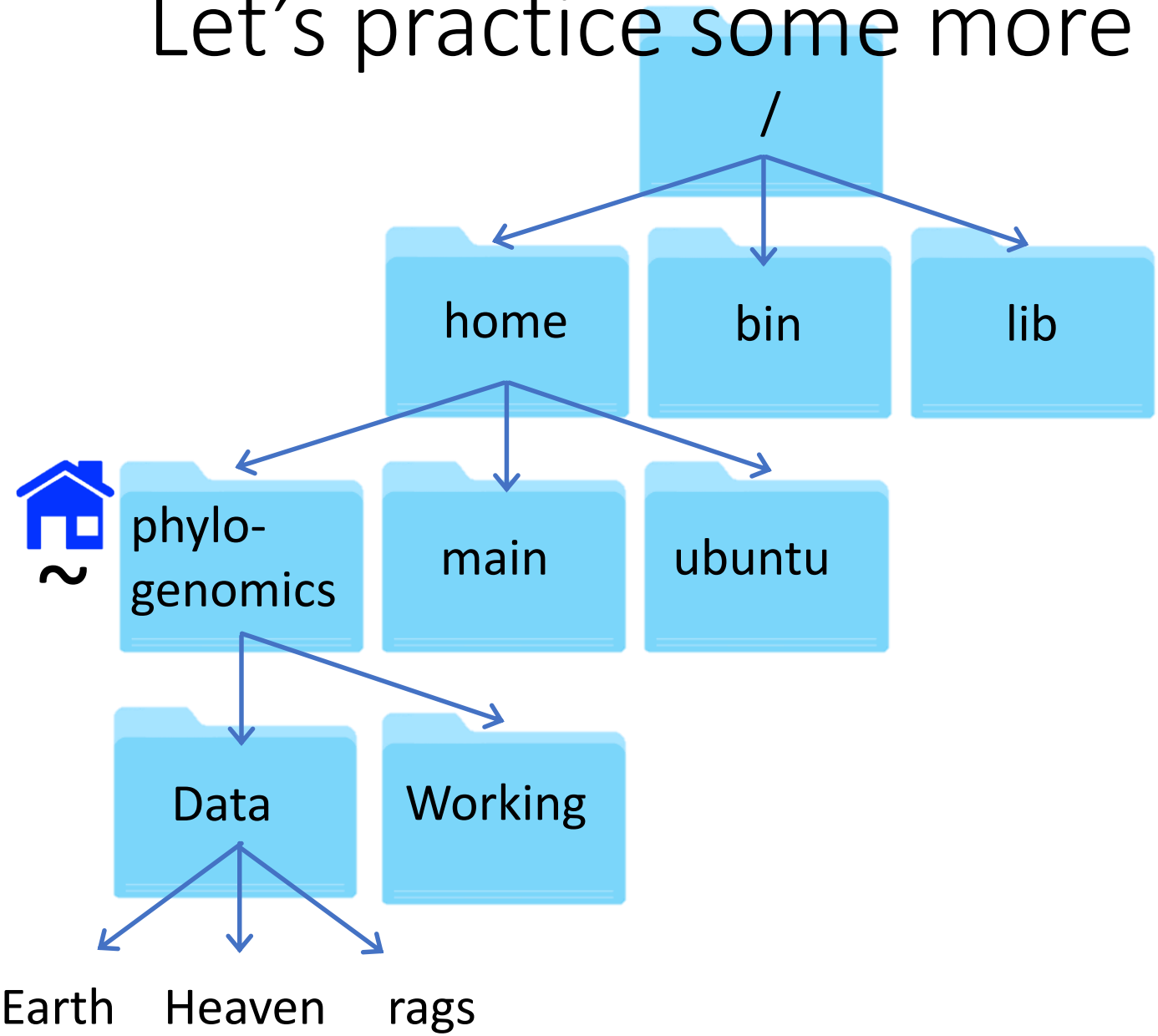
Let’s practice some more



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

Afterwards your file structure should look like this.

Let's practice some more



Moving Files

Lets move Heaven and Earth from Data to Working

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

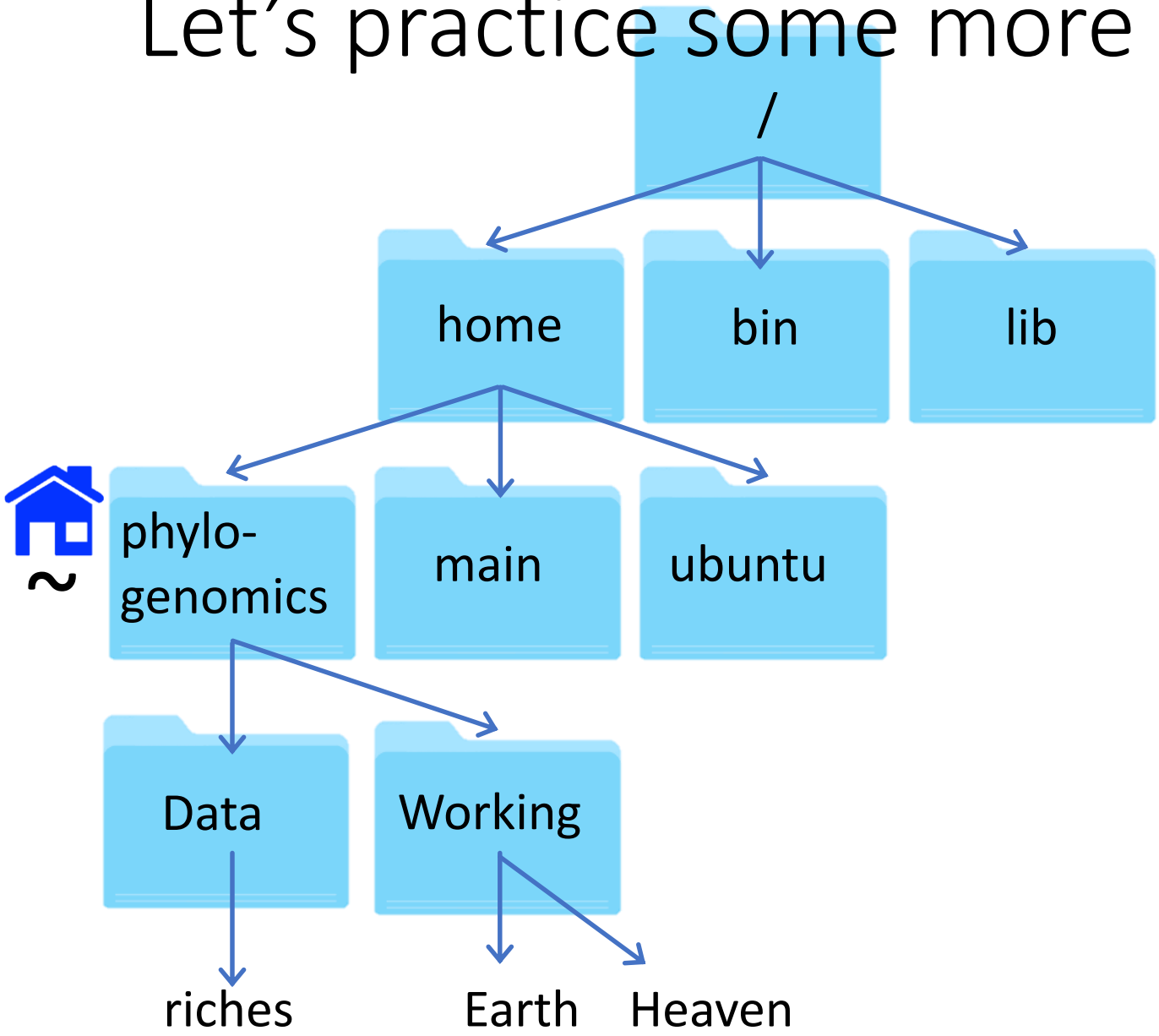
file path

destination path

Now move Earth, too.

Remember to Tab complete!

Let's practice some more



Moving Files

mv can also be used to rename files.

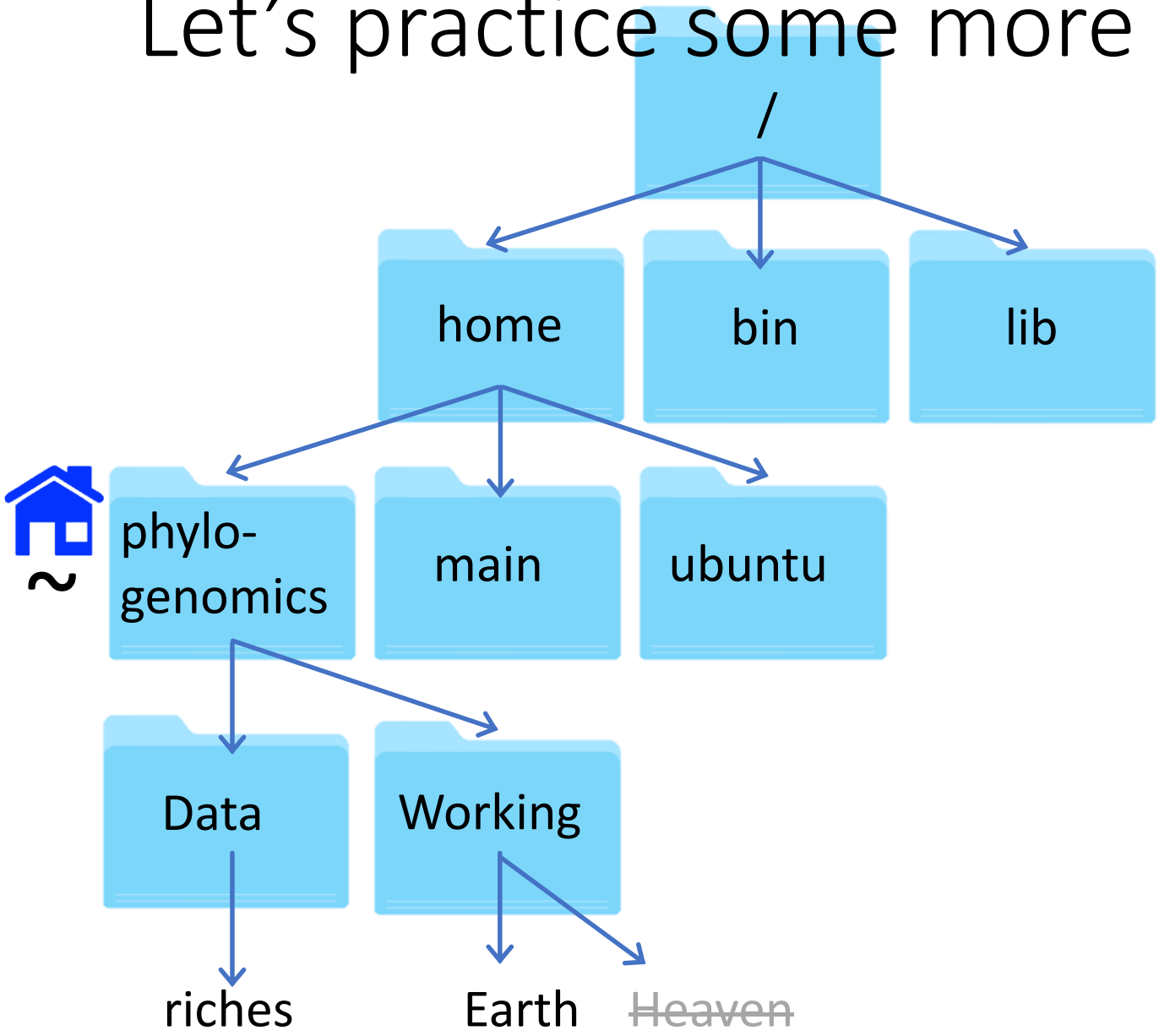
Let's change rags to riches.

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

file path

destination path

Let’s practice some more



Deleting Files

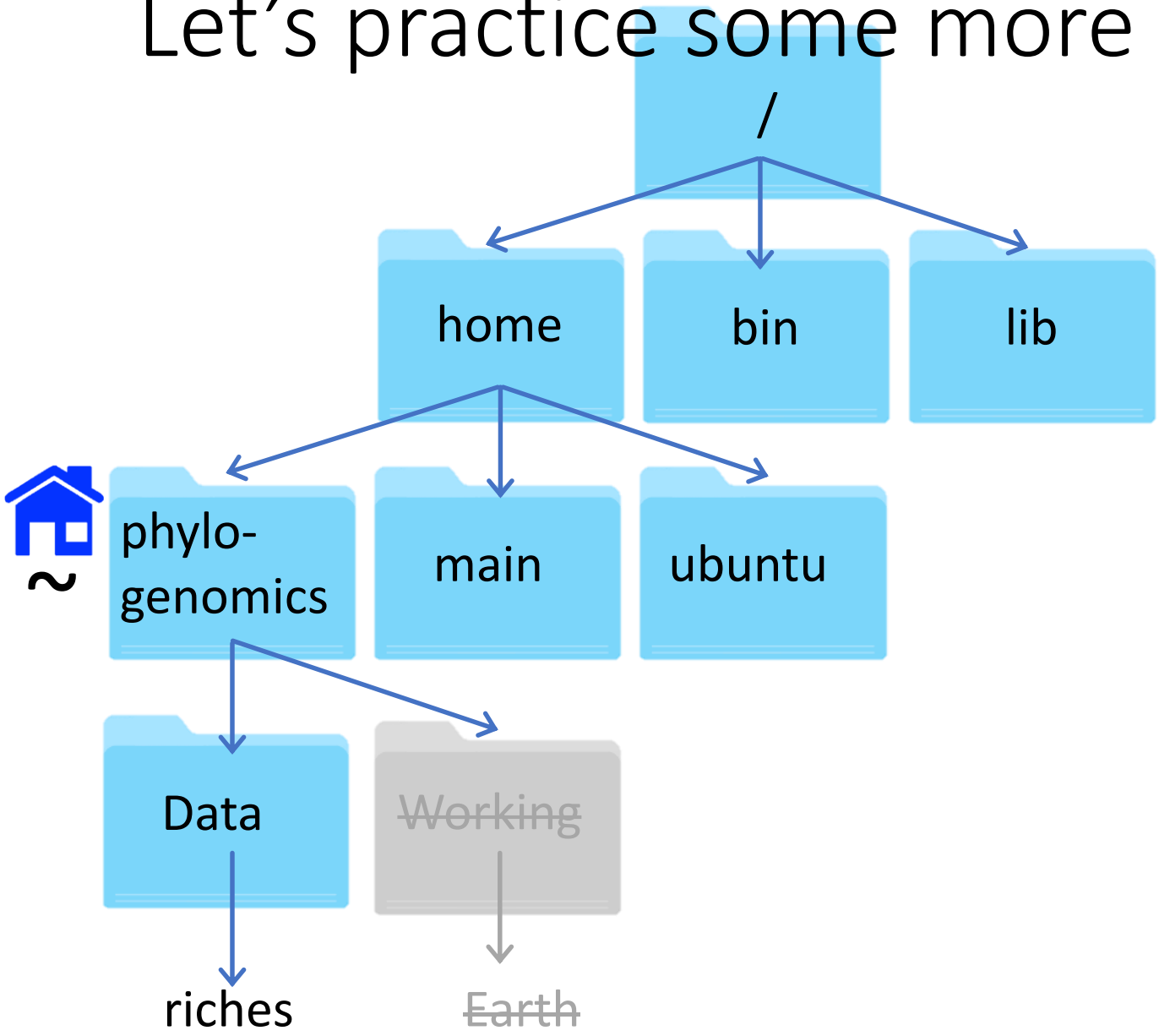
Now let’s delete Heaven

(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

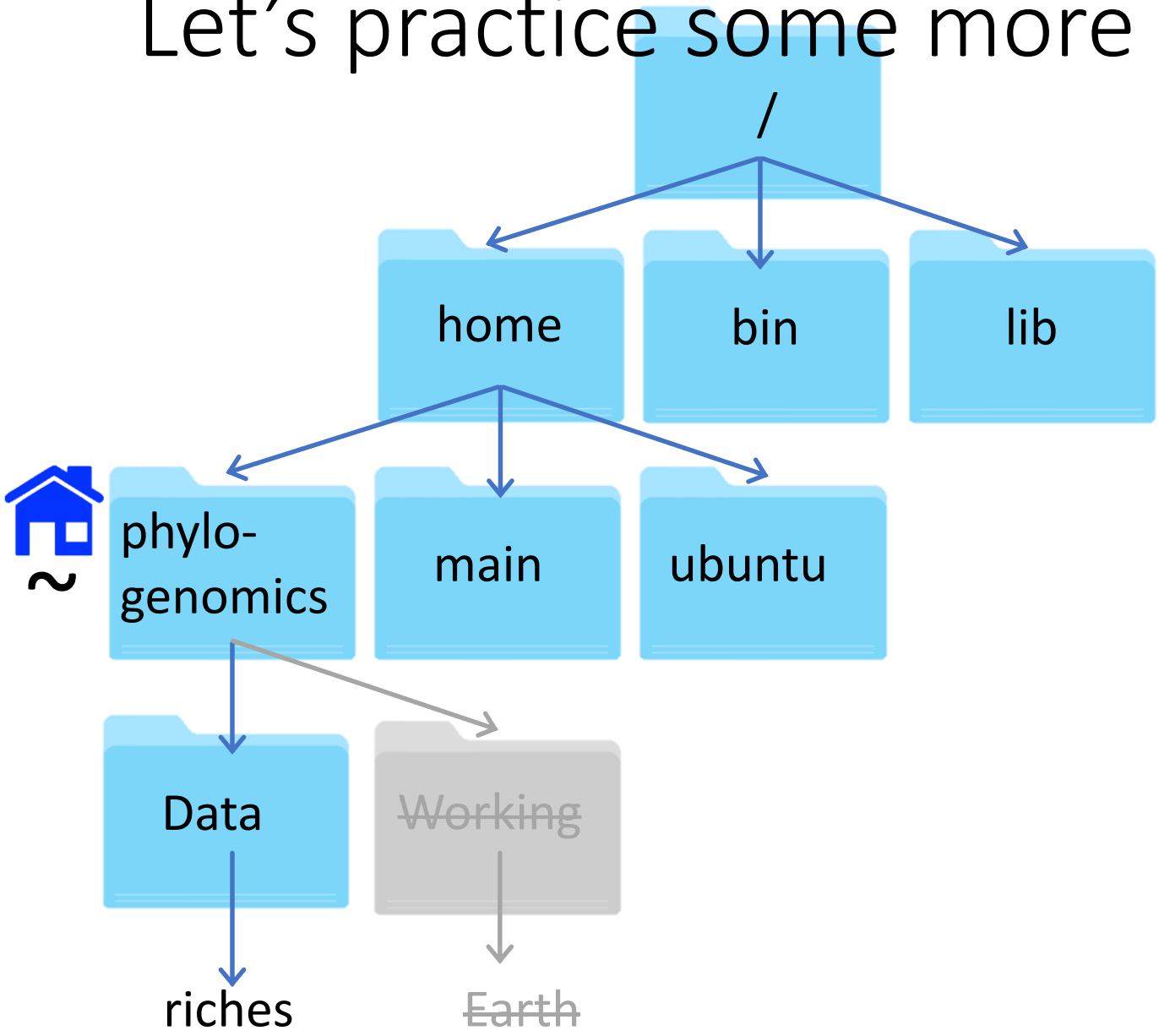


Deleting Files

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

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

Let’s practice some more



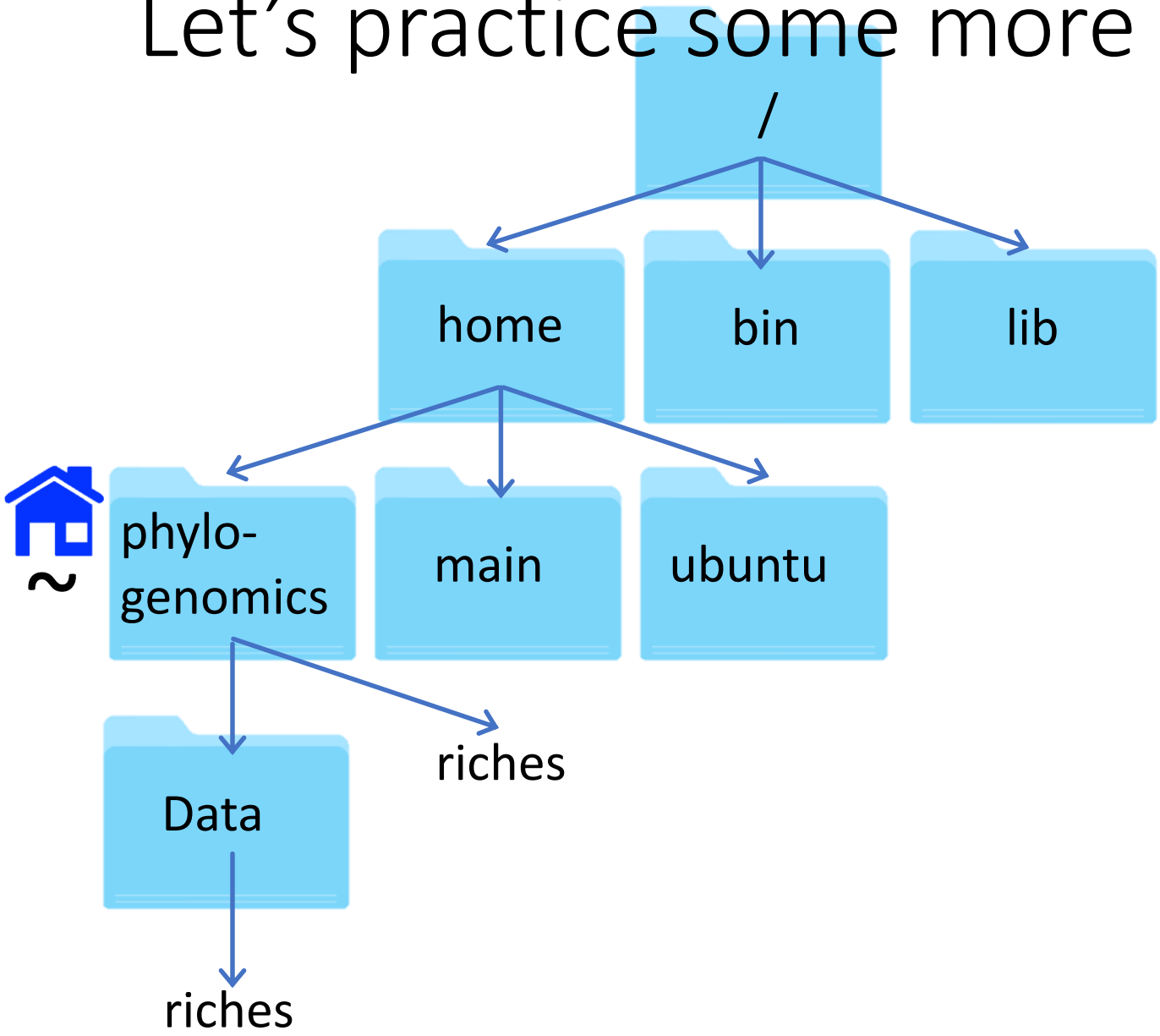
Deleting Files

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



Copying Files

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

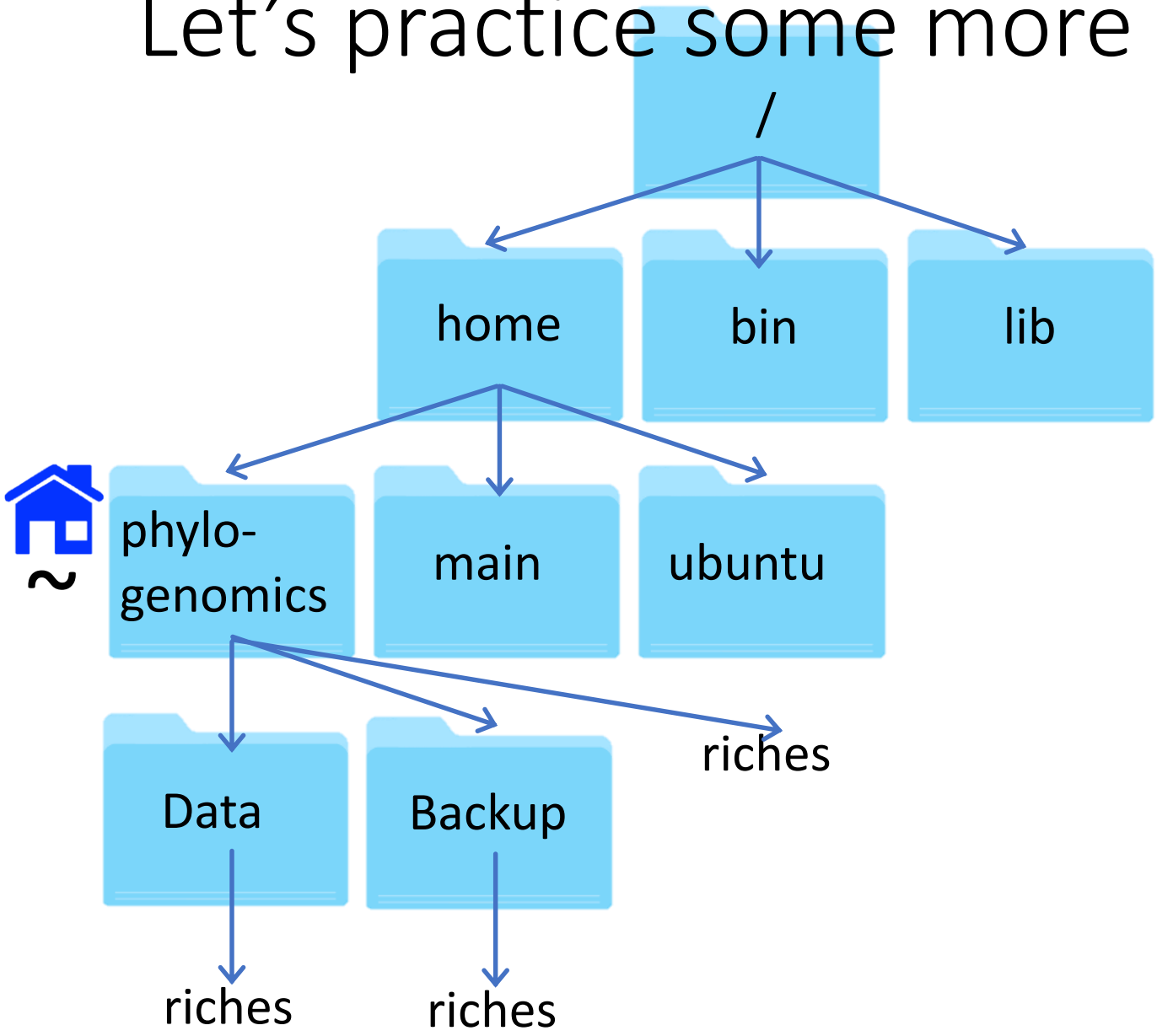
(Make sure your present working directory is Data.)

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

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 `$ cd ~`

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 uncompress 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

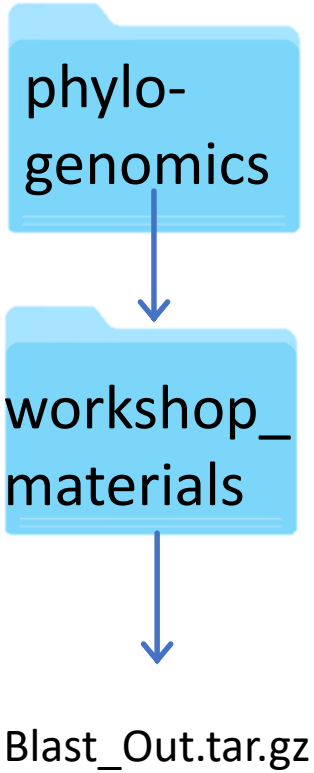
You should find a compressed directory: Sequences.tar.gz

2. Make a copy of this file in the Backup directory you created earlier
3. Un archive 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



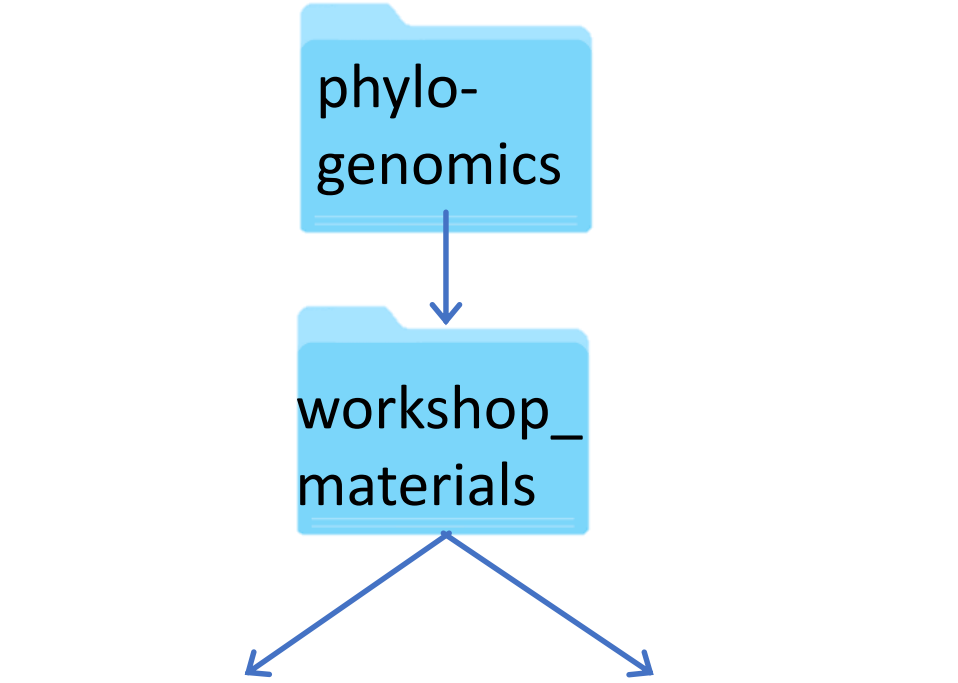
Navigate to the workshop_materials directory

```
$ cd ~/workshop_materials
```

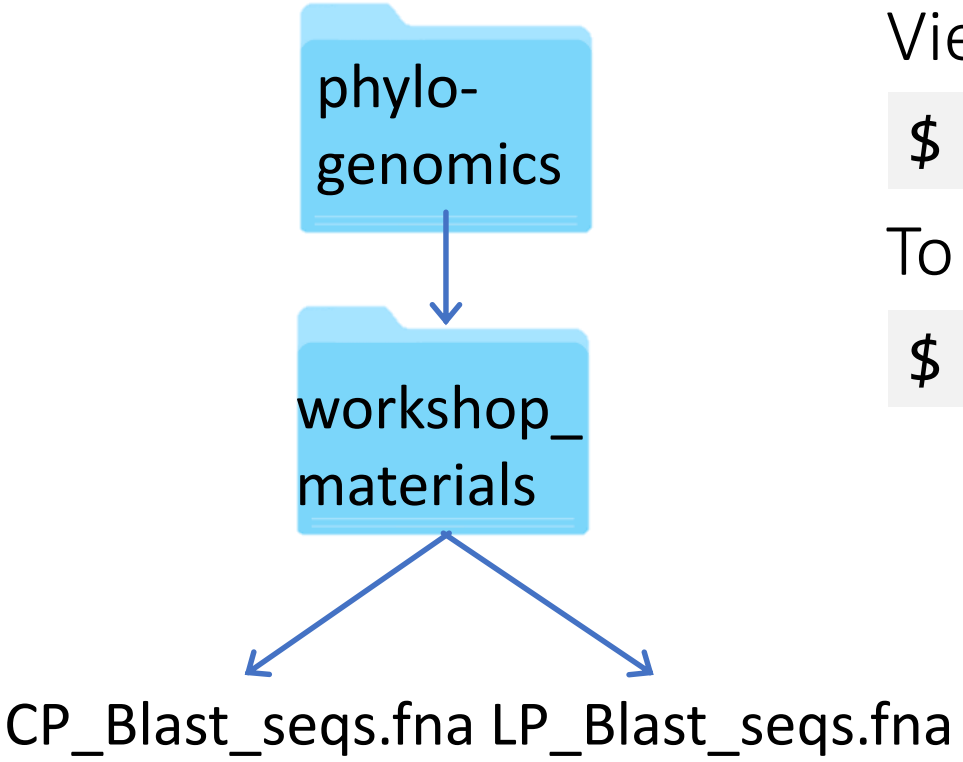
Unarchive the Blast_Out.tar.gz

```
$ tar -xzvf Blast_Out.tar.gz
```

Viewing and manipulating files



Viewing and manipulating files

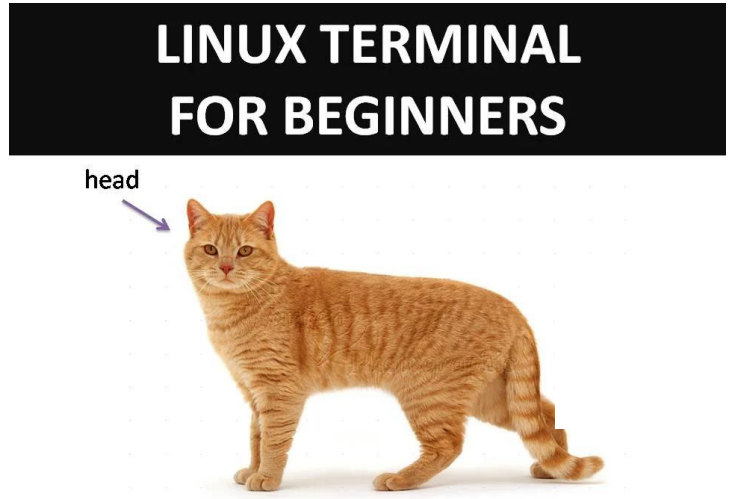


View the first 10 lines of a file

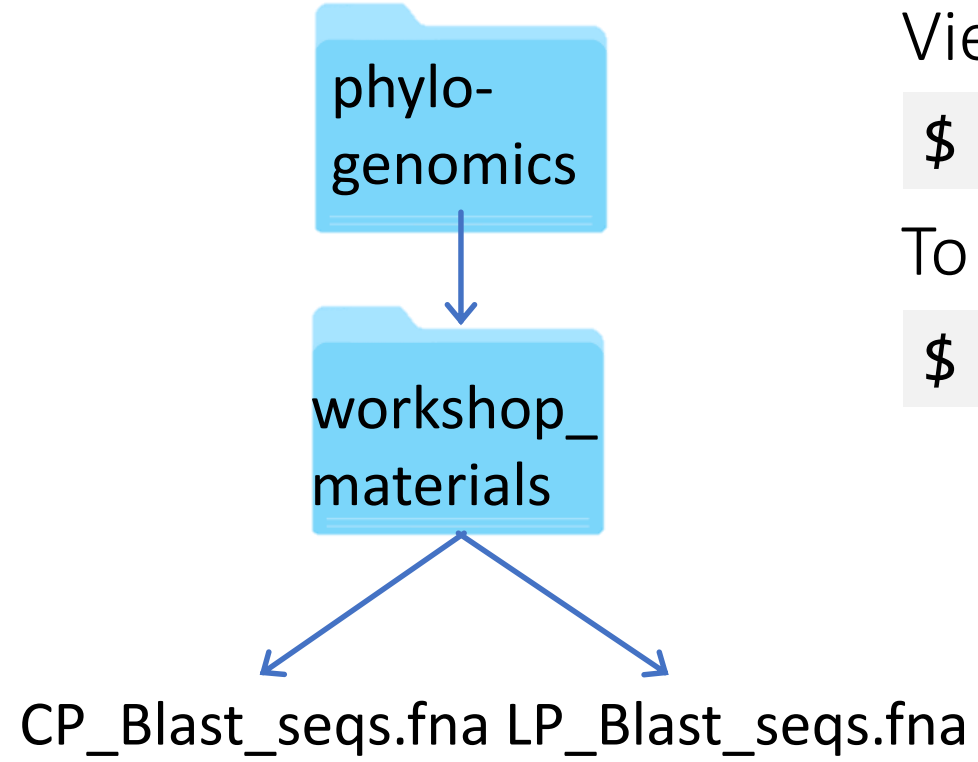
```
$ head CP_Blast_seqs.fna
```

To view the first 30 lines of the file

```
$ head -n 30 CP_Blast_seqs.fna
```



Viewing and manipulating files

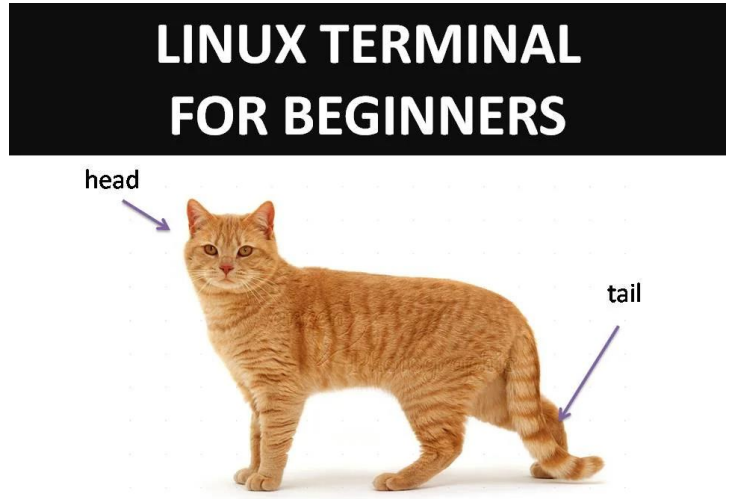


View the last 10 lines of a file

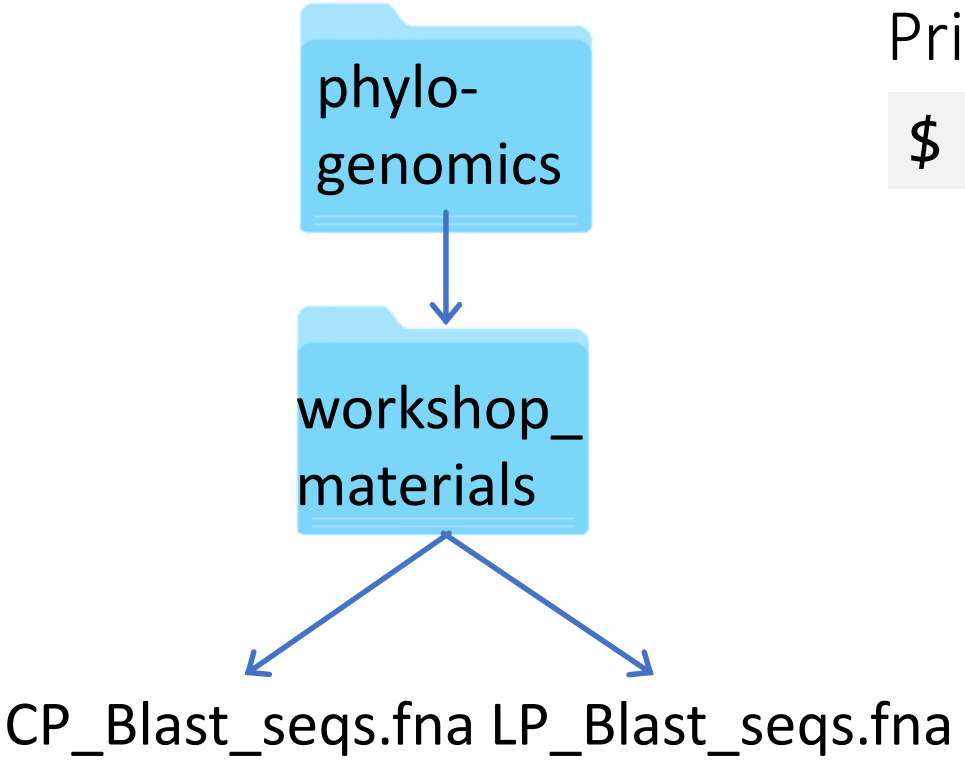
```
$ tail CP_Blast_seqs.fna
```

To view the last 30 lines of the file

```
$ tail -n 30 CP_Blast_seqs.fna
```

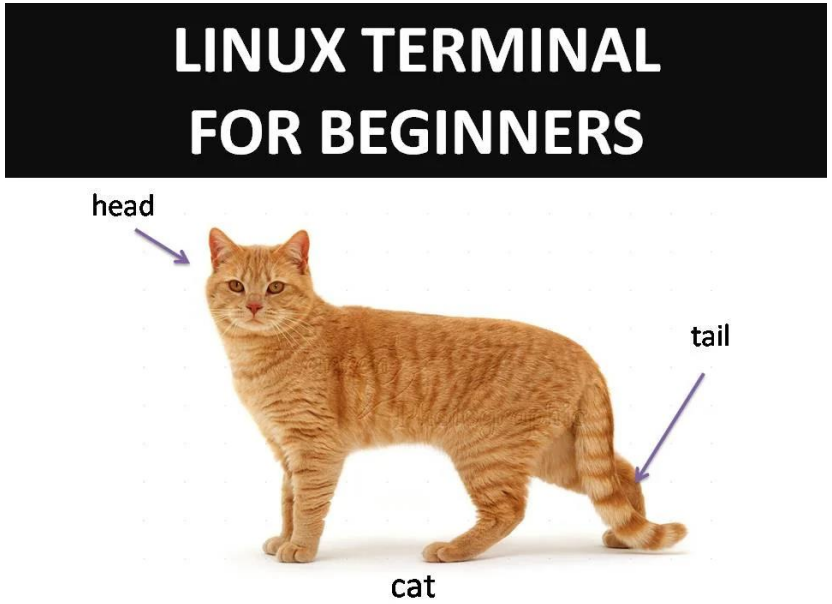


Viewing and manipulating files



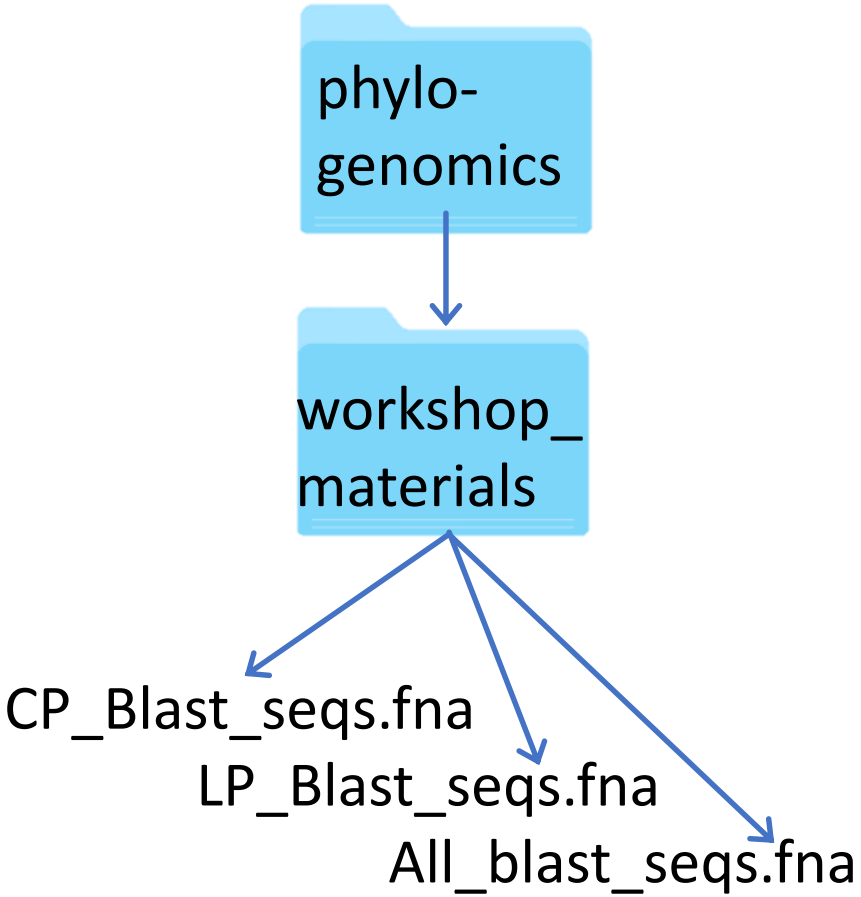
Print the entire file

```
$ 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



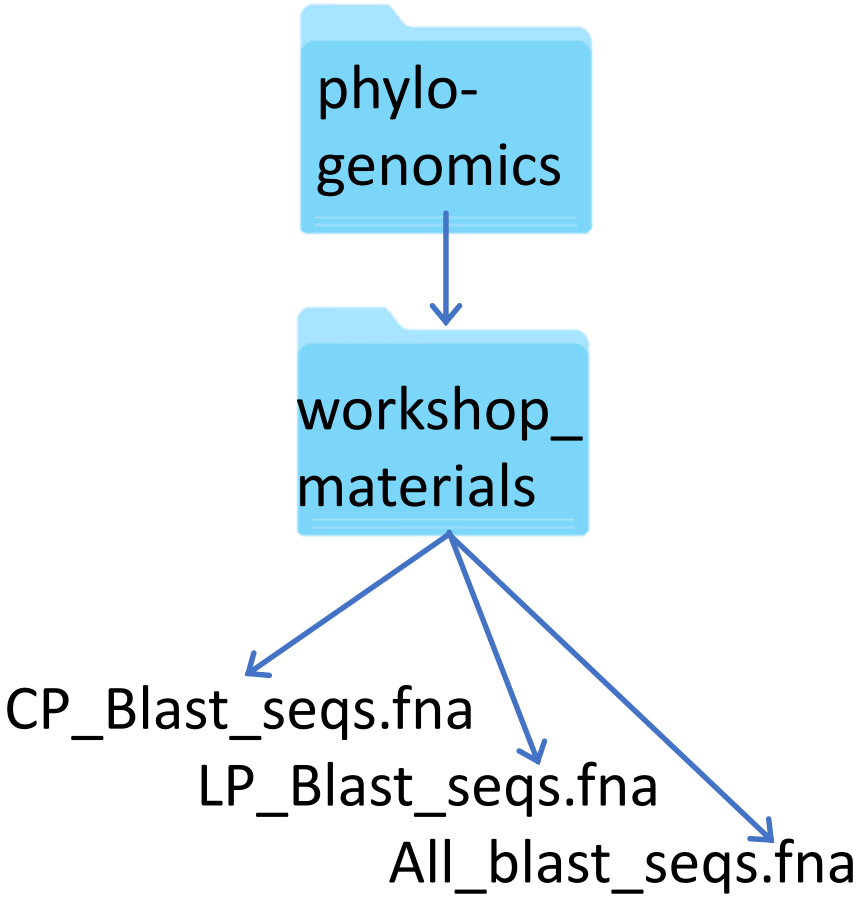
'cat' can also combine multiple files

```
$ 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
```

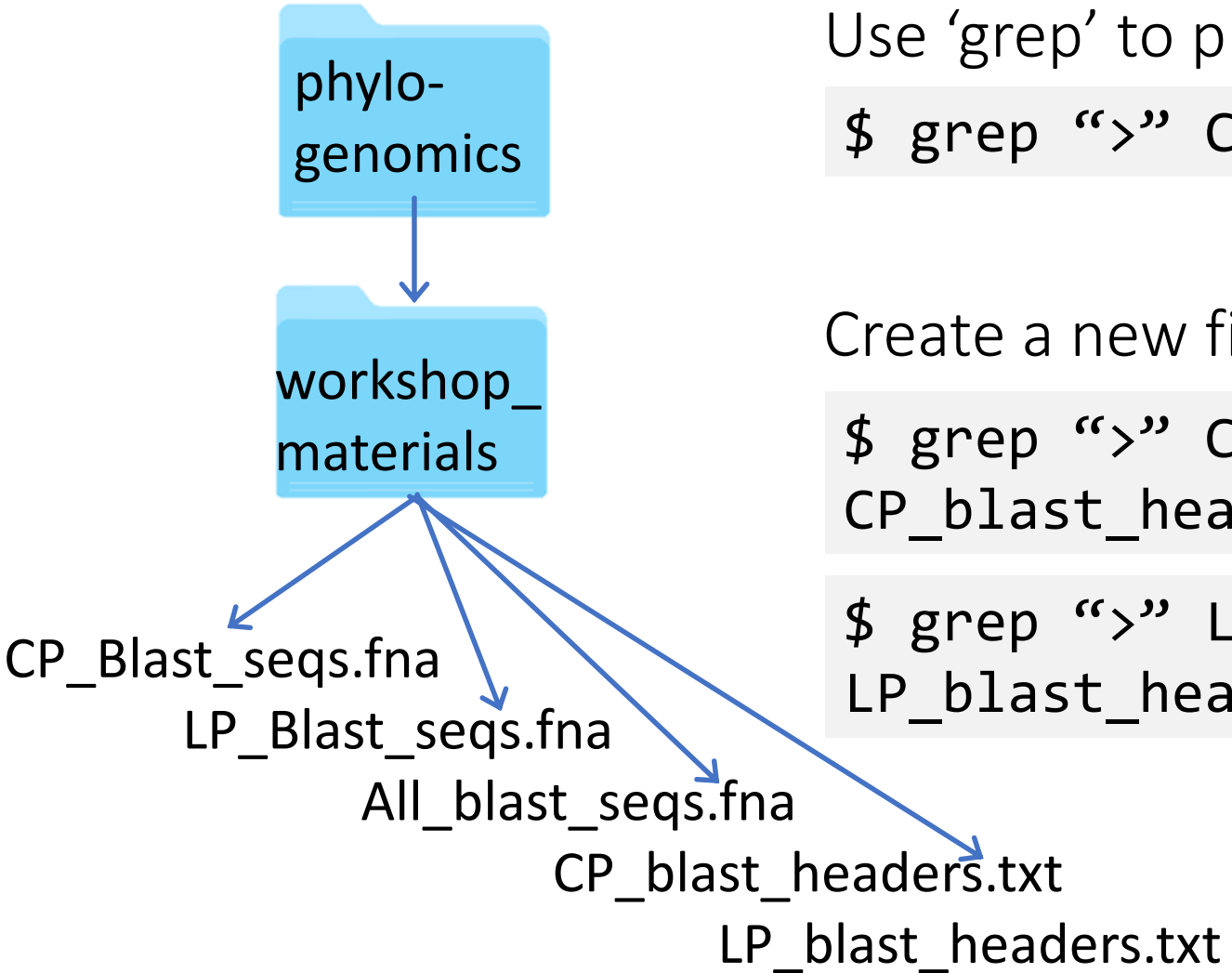
Viewing and manipulating files



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

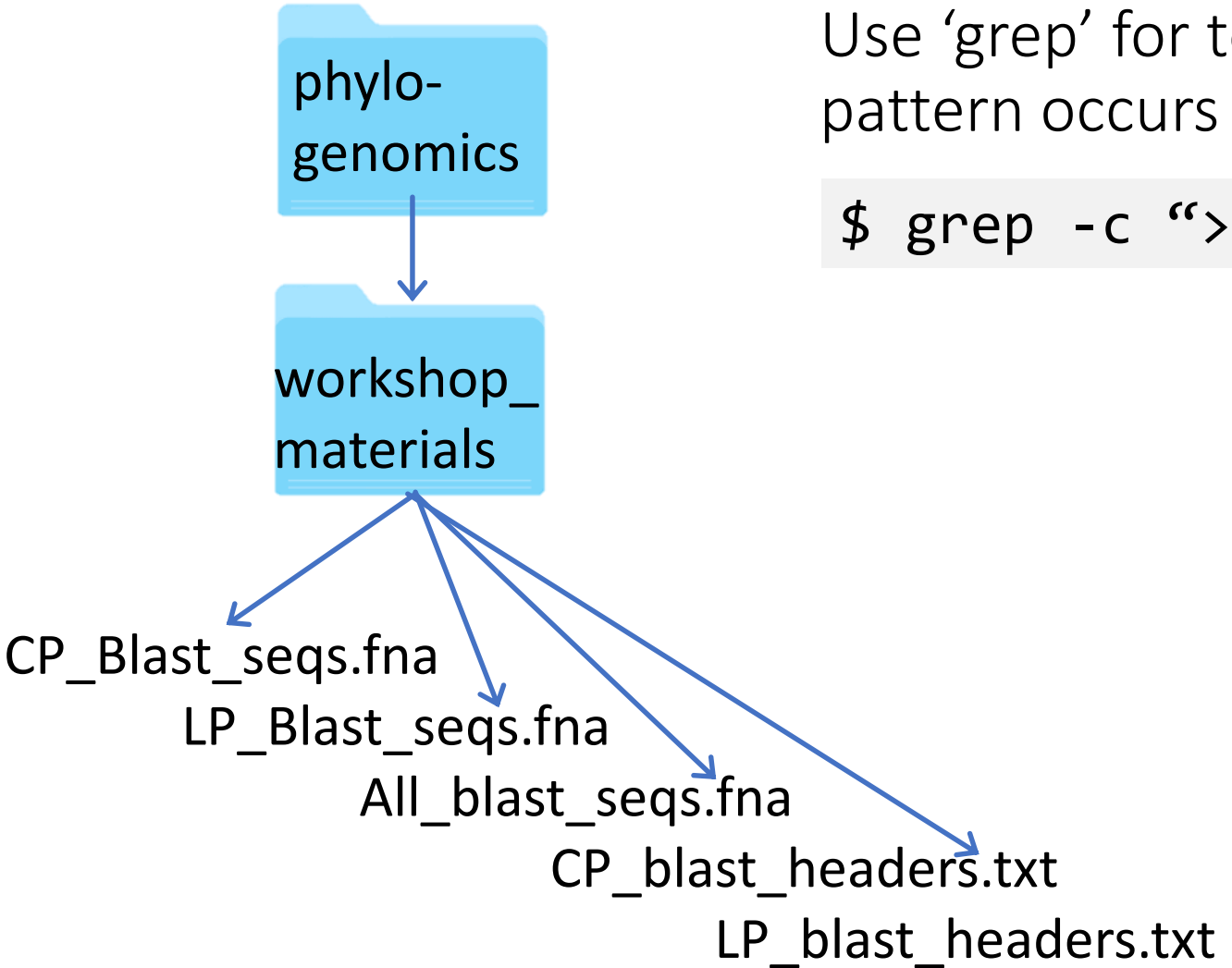
```
$ grep “>” CP_Blast_seqs.fna
```

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
```

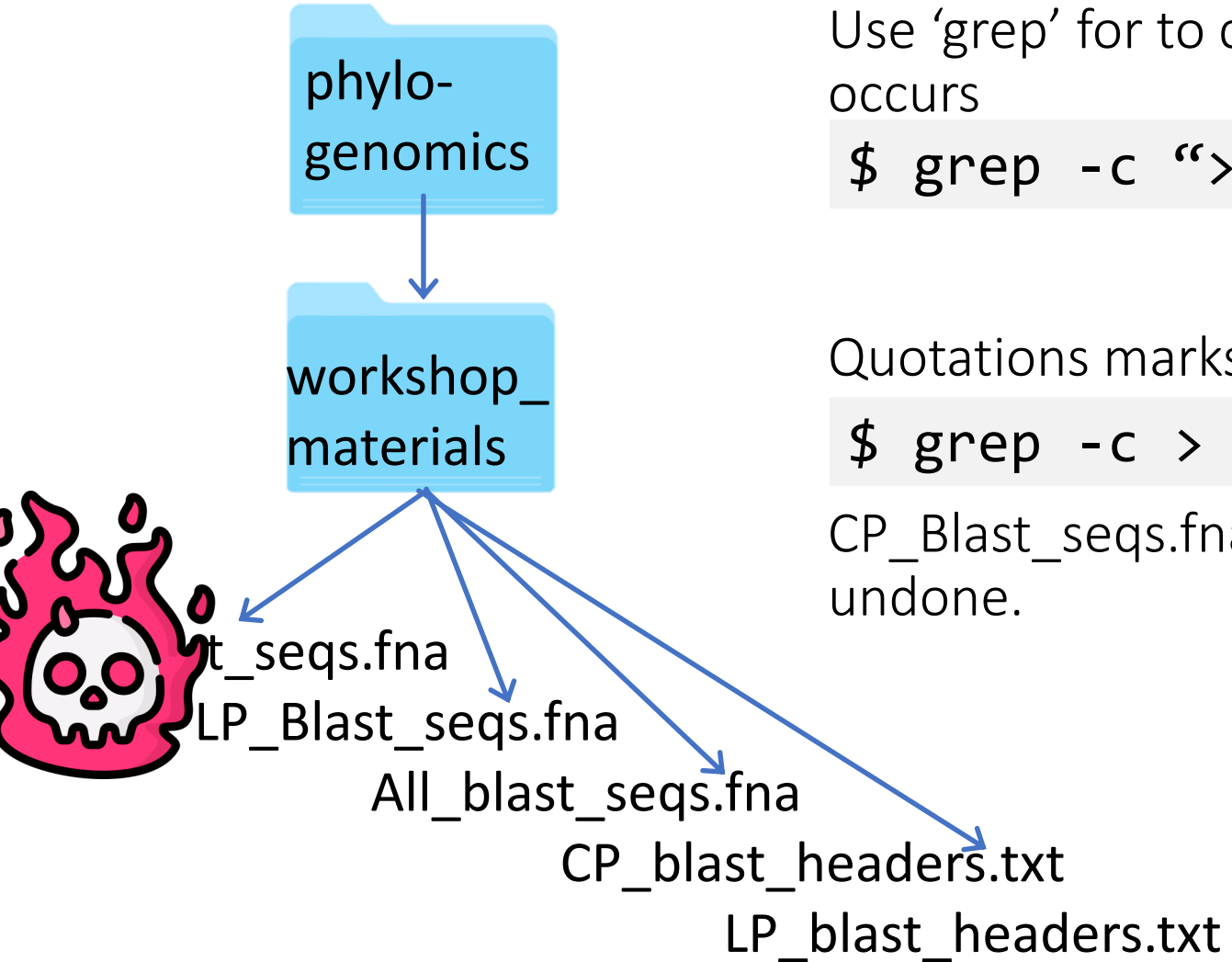
Viewing and manipulating files



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

```
$ grep -c “>” CP_Blast_seqs.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_seqs.fna
```

Viewing and manipulating files

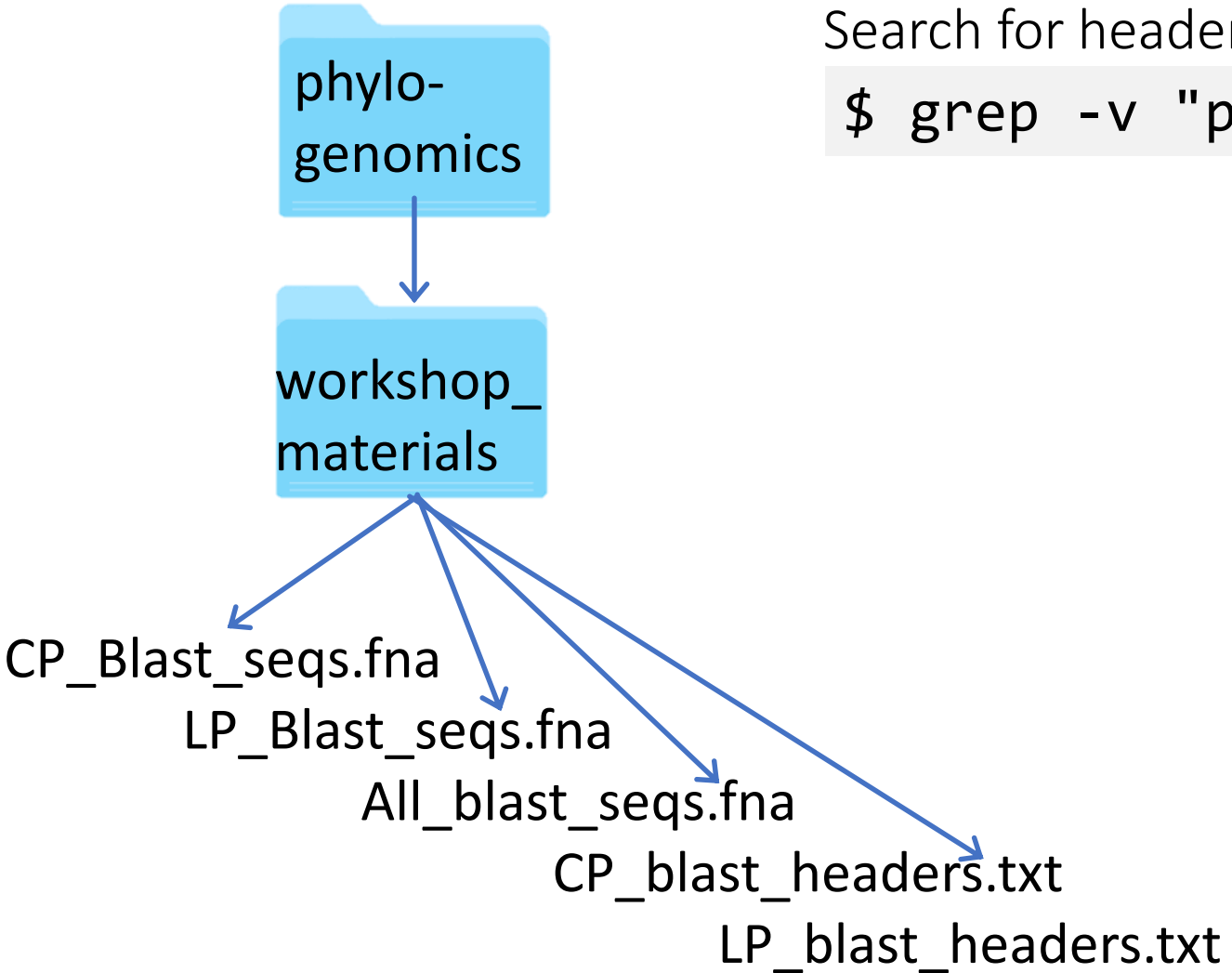
What can you do to your file to protect them? Change permissions

```
$ 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.

```
$ chmod +x my_script.sh
```

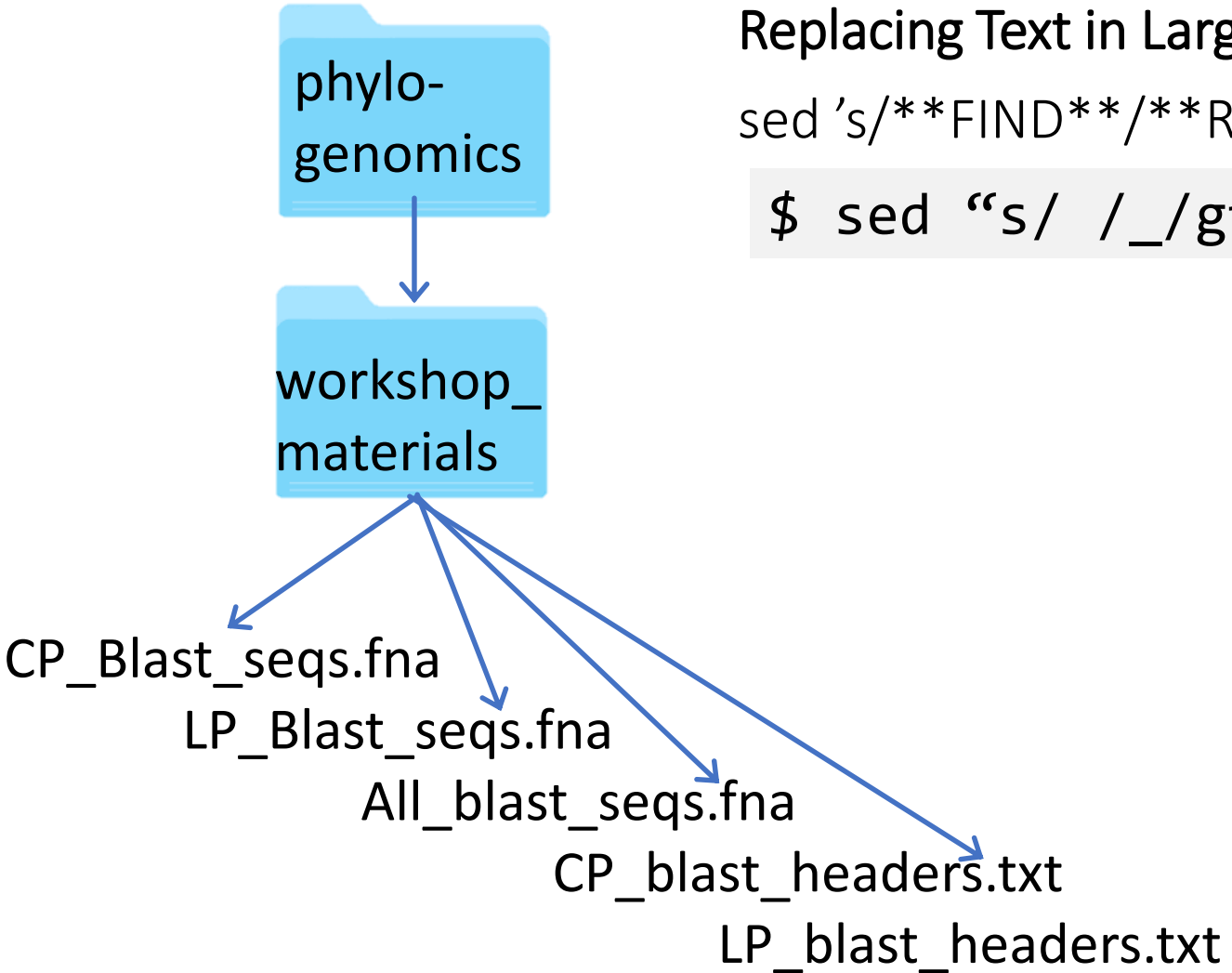
Viewing and manipulating files



Search for headers that are not partial sequences

```
$ grep -v "partial" LP_headers.txt
```

Viewing and manipulating files

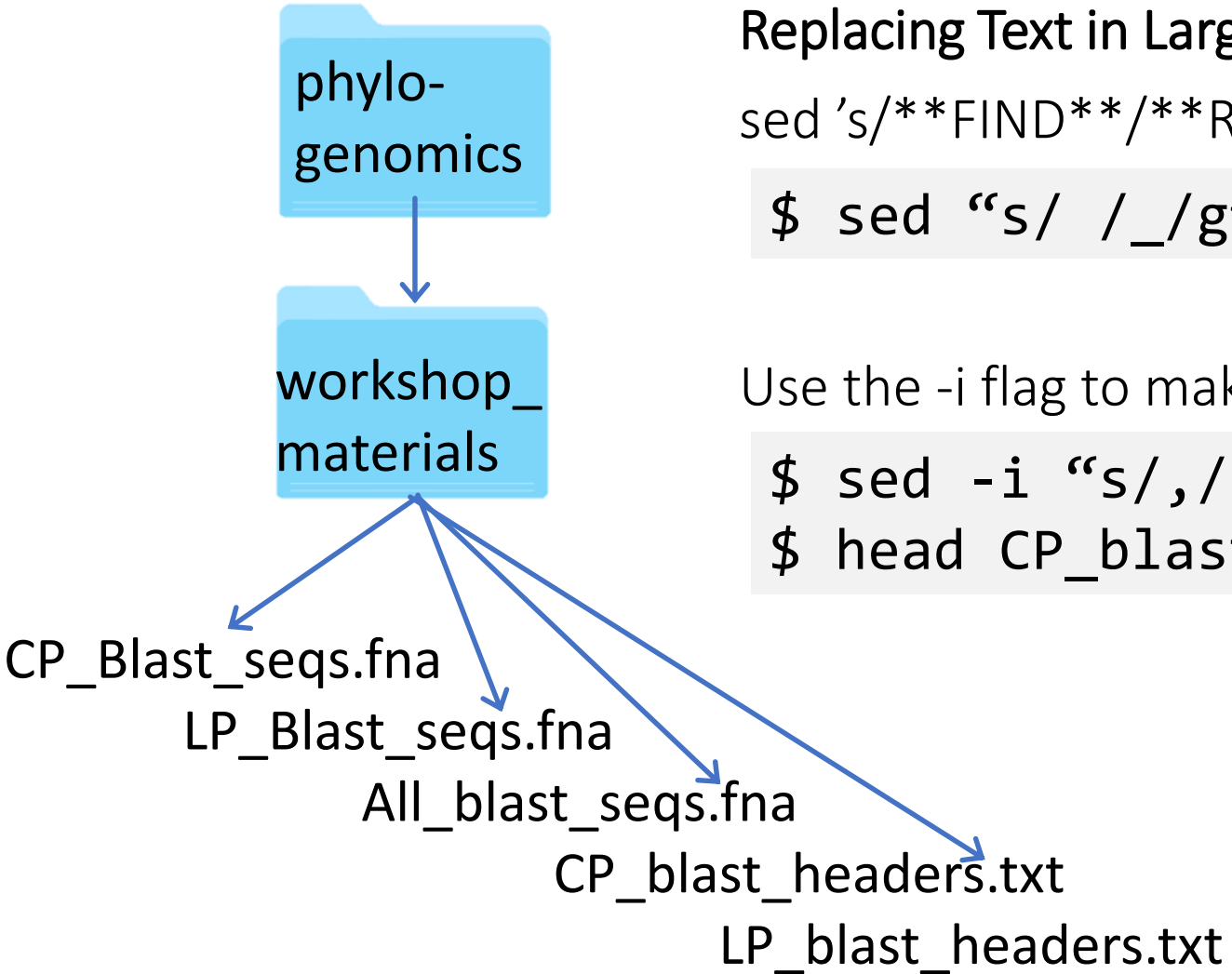


Replacing Text in Large files

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

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

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

Expression	Modern equivalent	Pattern matched
.		a single character
.+		one or more characters
.*		zero or more characters
.?		maybe present
^		first in the line
$		last in the line
[0-9]	\d	digits
[a-zA-Z]	\w	letters
‘ ‘	\s \t	space
{2}		exactly 2 characters long
{2,4}		between 2 and 4 characters long
[ACGT]		a specific set of characters

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 |

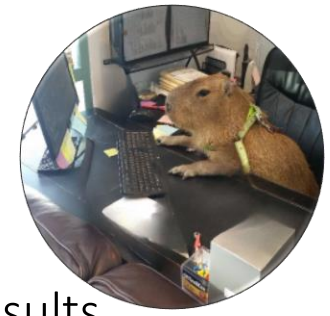
Combine the fasta files from the blast output and identify how many sequences they contain

```
$ 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.

```
$ 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”.

```
$ for i in *.fna; do grep -c “CAT” $i; done
```

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

```
$ 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,

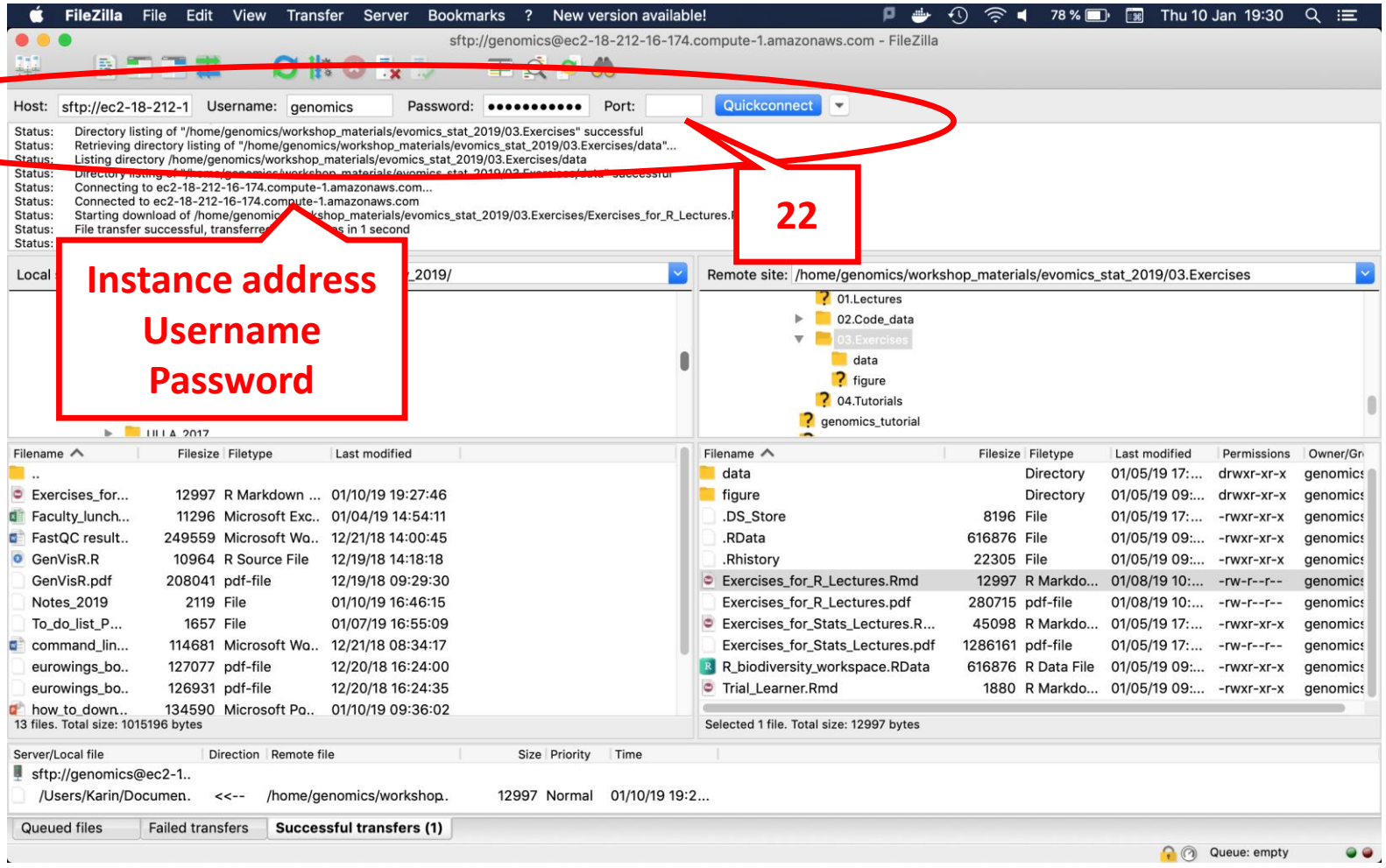
scp –r phylogenomics@<your.IP.address>:/Location/On/Instance /Local/path

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

```
$ rsync –avz phylogenomics@ec … .
```

Downloading

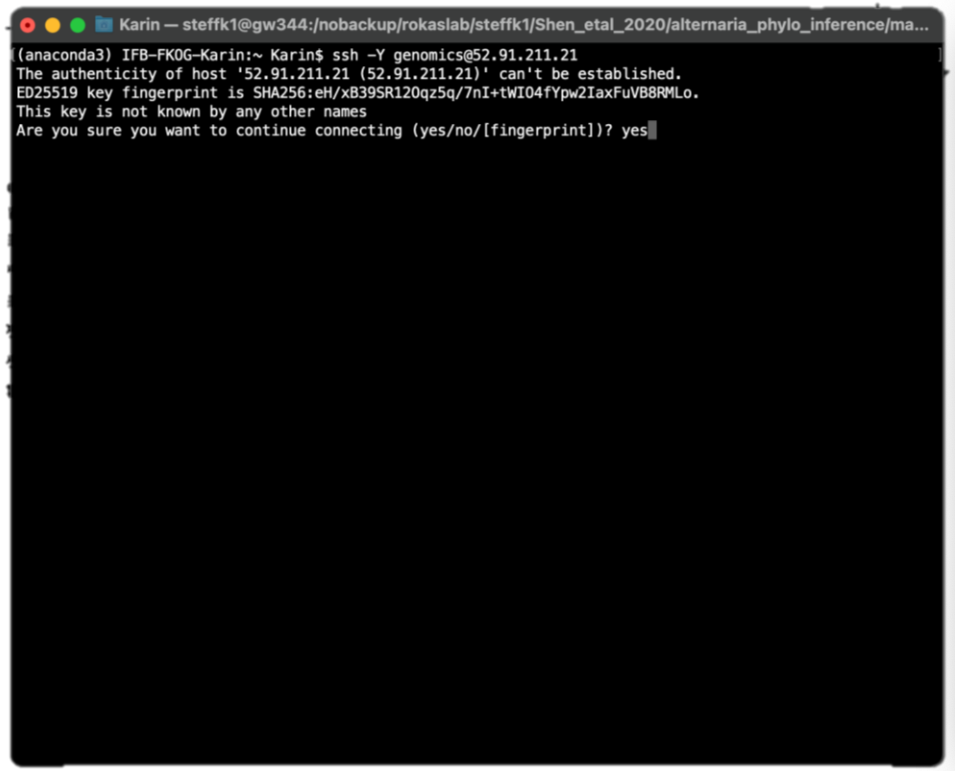
GUI: Filezilla https://sourceforge.net/projects/filezilla/

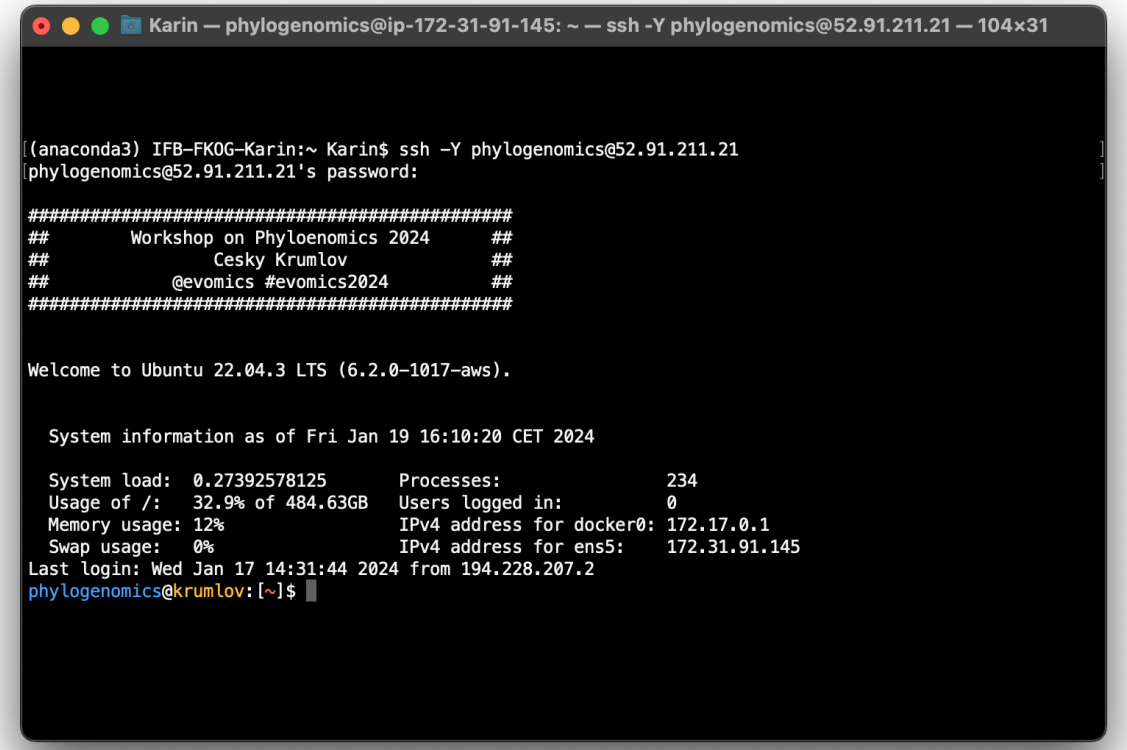


Logging in to the instance through your local terminal

`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!)