

Workshop 1:

Introduction to UNIX command-line

Peter Scott, PhD | pscott17@ucla.edu

QCB Fellow



"Swiss Army knife" set of tools

Thanks to Serghei Mangul for base slides!

Peter Scott, PhD

pscott17@ucla.edu

Shaffer Lab

<https://sites.lifesci.ucla.edu/eeb-shafferlab/>

Department of Ecology and Evolutionary Biology

University of California, Los Angeles



Collaboratory Website

<http://qcb.ucla.edu/collaboratory/>

Workshop 1: Introduction to UNIX command--line

- **Day 1**
 - Unix - Learning the essentials
 - Unix fundamentals, syntax, and usage
- **Day 2**
 - Unix commands
 - Useful tools for processing text files
- **Day 3**
 - Useful shell commands
 - UNIX Shell Scripting
 - Running jobs on the Hoffman2 cluster

Why Unix?

- As biological data sets have grown **larger** and biological problems have become more complex, the requirements for computing power have also grown.
- Computers that can provide this power generally use a Unix/Linux operating system (e.g. Hoffman2)



Why Unix?



- It is **very popular**, so it is easy to find information and get help
- Unix is very **stable** – computers running Unix almost never crash
- Unix is very **efficient**
 - manage extremely **huge amounts of data**
- Most new bioinformatics software is created for Unix

Command line interface

Topic	CLI	GUI
Ease	Due to a higher degree of memorization and familiarity needed for operation and navigation, new users find operating a command line interface more difficult than a GUI.	Because a GUI is much more visually intuitive, new users almost always pick up this interface faster than a CLI.



Command line interface

Topic	CLI	GUI
Control	Users have more control over both the file and operating systems in a command line interface. For example, users can copy a specific file from one location to another with a one-line command.	Although a GUI offers ample access to the file and operating systems, advanced tasks may still need to utilize the command line.
		

Command line interface

Topic	CLI	GUI
Multitasking	Although many command line environments are capable of multitasking, they do not offer the same ease and ability to view multiple things at once on one screen.	GUI users have windows that enable a user to view, control, manipulate, and toggle through multiple programs and folders at same time.



Command line interface

Topic	CLI	GUI
Speed	Command line users only need to utilize their keyboards to navigate a the interface. Additionally, they oafen only need to execute a few lines to perform a task.	Using both a mouse and keyboard to navigate and control your operating or file system is going to be much slower than someone who is working in a command line.





Command line interface

Topic	CLI	GUI
Scripting	A command line interface enables a user to script a sequence of commands to perform a task or execute a program.	Although A GUI enables a user to create shortcuts, tasks, or other similar actions, it doesn't even come close in comparison to what is available through a command line.



Command line interface

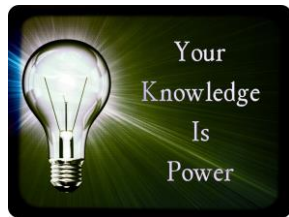
Topic	CLI	GUI
Diversity	After you've learned how to navigate and use a command line, it's not going to change as much as a new GUI. Although new commands may be introduced, the original commands always remain the same.	Each GUI has a different design and structure when it comes to performing different tasks. Even different iterations of the same GUI, such as Windows, can have hundreds of different changes between each version.
		

Command line interface

Topic	CLI	GUI
Strain	The command line allows the user to keep their hands on the keyboard, almost never touching the mouse. Moving back and forth between a keyboard and mouse can cause additional strain and may help contribute to Carpal Tunnel Syndrome .	Although shortcut keys can help reduce the amount of times you have move from the keyboard to the mouse, you will still be moving much more between devices in a GUI.



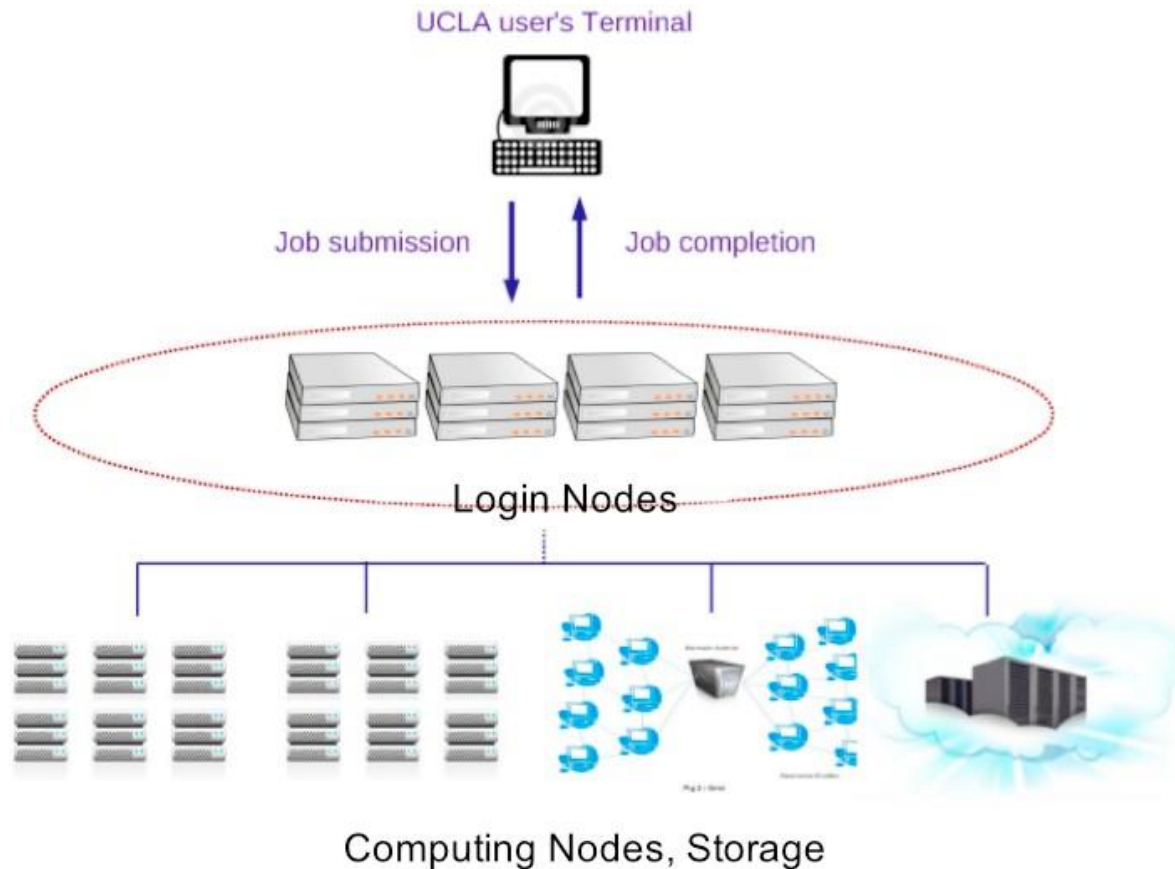
Do biologists need to become programmers?



*provided in the
class

*free and easy to use

Hoffman2



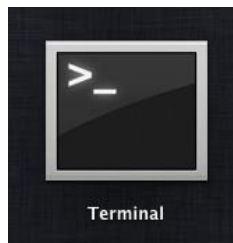
How to connect to hoffman2

- Open a **SSH** program on your computer
- Connect to: **hoffman2.idre.ucla.edu**
- Type your **username** and **password**
 - ssh pscott17@hoffman2.idre.ucla.edu
 - pscott17@hoffman2.idre.ucla.edu's
password:
 - Notice that when you type a password, nothing shows up on the screen, this is for your security

Open SSH program



Terminal



putty, Cygwin, Ubuntu for
Windows (new app Windows 10)



<http://www.chiark.greenend.org.uk/~sgtatham/puKy/download.html>

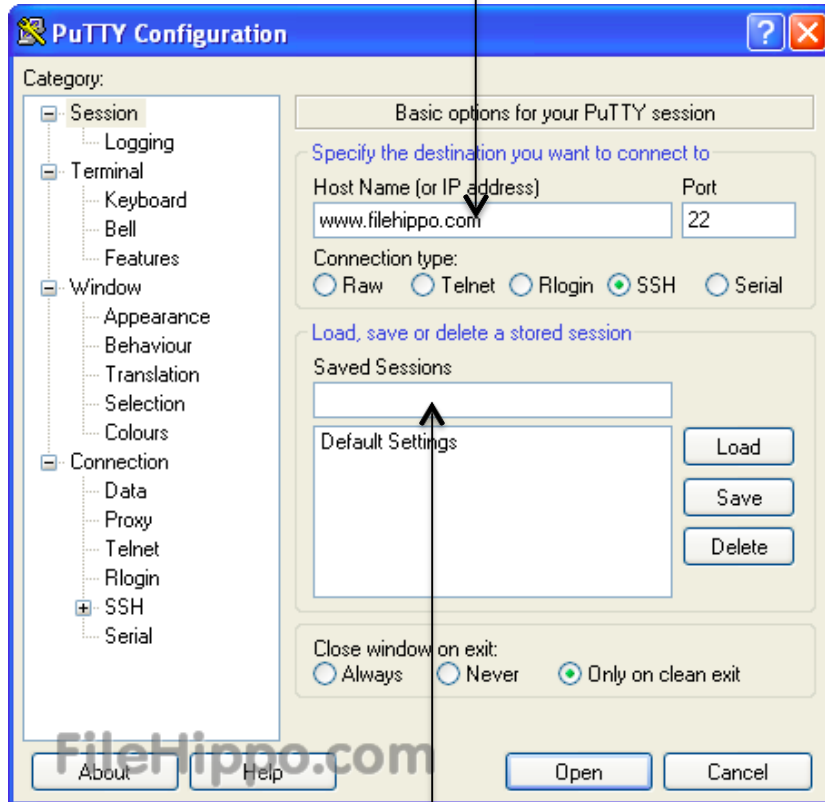


For Windows on Intel x86
PuTTY: [putty.exe](#)

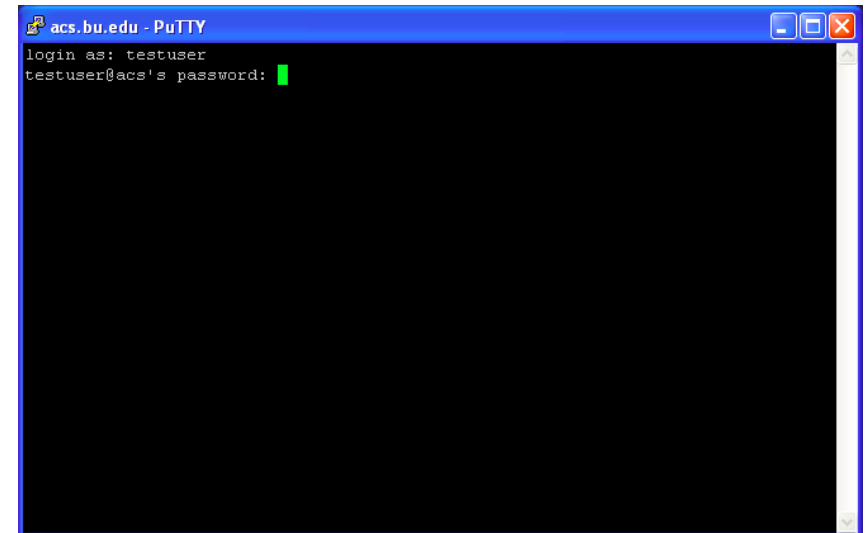


Connect to hoffman2

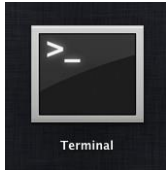
hoffman2.idre.ucla.edu



Session name (e.g. hoffman2)



"Yes" for fingerprint for the first time!



Connect to hoffman2

`ssh pscott17@hoffman2.idre.ucla.edu`



```
pscott17@login2:~  
Peter@DESKTOP-5HT6KUE ~  
$ ssh pscott17@hoffman2.idre.ucla.edu  
pscott17@hoffman2.idre.ucla.edu's password:  
Last login: Mon Jun 24 14:37:01 2019 from wifi-131-179-38-207.host.ucla.edu  
(base) [pscott17@login2 ~]$ |
```

The Unix Shell



- A shell is a program that waits for you to type a command and then executes it.
 - type the command, then “return”

The Unix Shell



- A shell is a program that waits for you to type a command and then executes it.
 - Uses a general basic syntax:
 - “program/utility/language” +flags +file

The Unix Shell



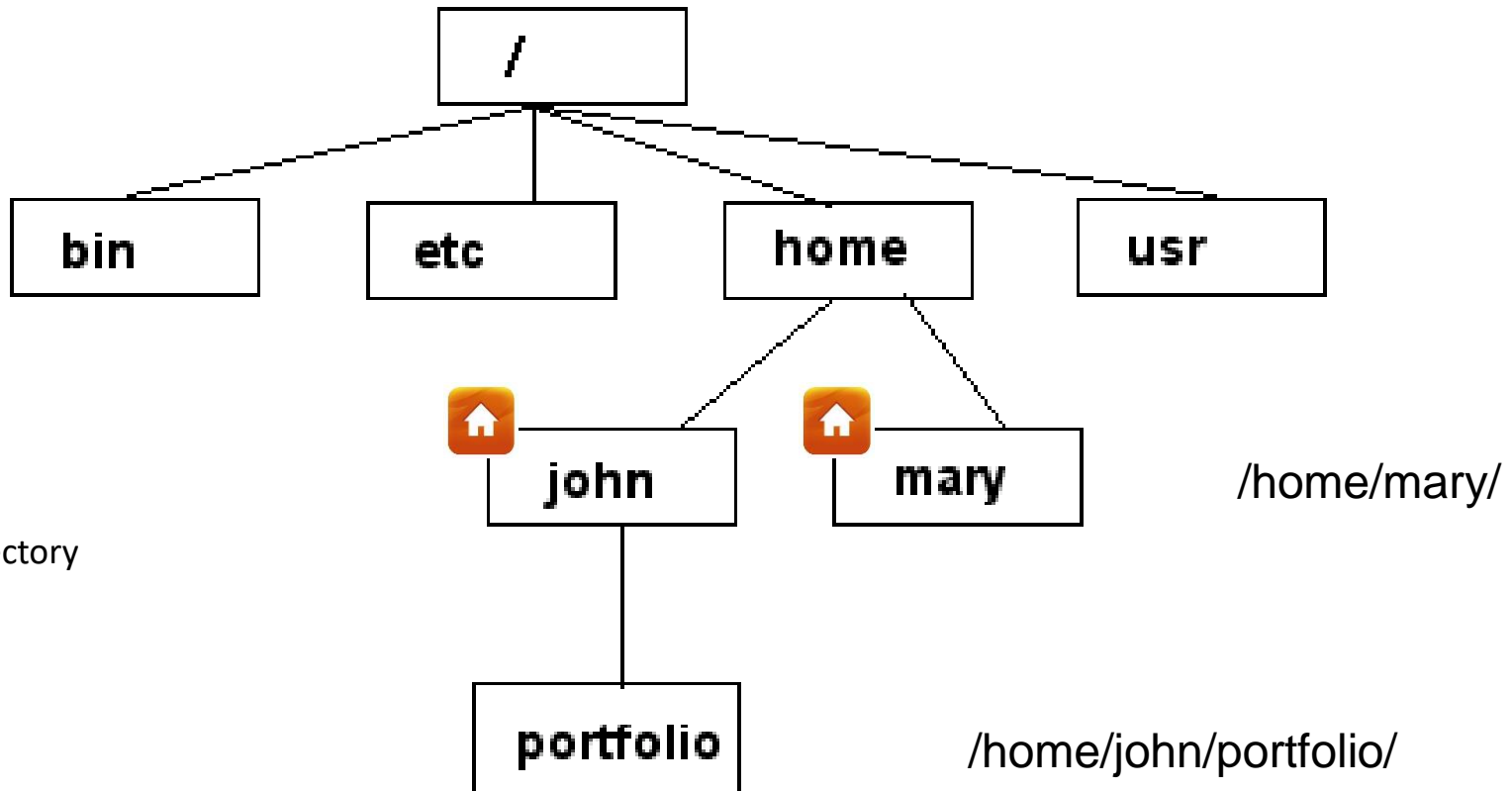
- A shell is a program that waits for you to type a command and then executes it.
 - Uses a general basic syntax:
 - “program/utility/language” +flags +file

```
$ ls -l *.txt
```

Translates to: list (`ls`); modify display to long (`-l`);
all text files (`*.txt`)

Unix File System

Unix is cAsE sEnsItiVe !



Home directory

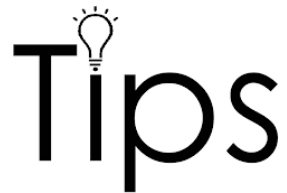
Hoffman2 : `/u/home/p/pscott17/project/`

my home directory

↑
path

Home directory

- When you login to the hoffman2 server, you always start in your **Home** directory.
- Create sub-directories to store specific projects or groups of information



Do **not** accumulate thousands of files with cryptic names in your Home directory

Command: passwd

- changes your hoffman2 password
- A very good idea after you got a default one.

```
[psscott17@login3 ~]$ passwd
```

```
Changing password for user psscott17.
```

```
Please enter your current password:
```


Command: pwd

- To display current directory

```
[psscott17@login3 ~]$ pwd  
/u/home/p/psscott17
```

Command: mkdir

- To create a new directory use “mkdir”

```
[psscott17@login3 ~]$ mkdir test
```



If no error message is displayed
means the command was run
successfully

Command: cd

- cd changes your current working directory

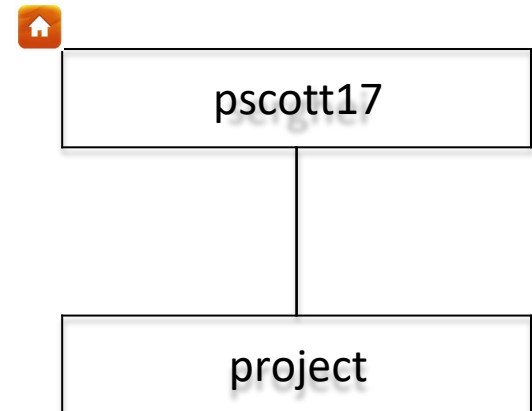
```
[psscott17@login3 ~]$ cd test  
[psscott17@login3 test]$ pwd  
/u/home/p/psscott17/test
```

Command: cd

- “~” is the location of your home directory
- “..” is the location of the directory above the current one

my home directory

/u/home/p/pscott17/project/



Let's practice



```
[psscott17@login3 test]$ cd ~
```

```
[psscott17 @login3 ~]$ pwd  
/u/home/p/psscott17
```

```
[psscott17 @login3 ~]$ cd ..  
[psscott17@login3 s]$ pwd  
/u/home/p
```

```
[psscott17@login3 s]$ cd psscott17/test  
/u/home/p/psscott17
```

T[💡]ips

- to go back to previously entered commands, use the **up** and **down** arrows
- to auto--complete file names, use the **tab** key
- if you are **stuck** within a command/process/program, try **ctrl-z (Mac)** or **ctrl-c (Linux/Windows)** to terminate it



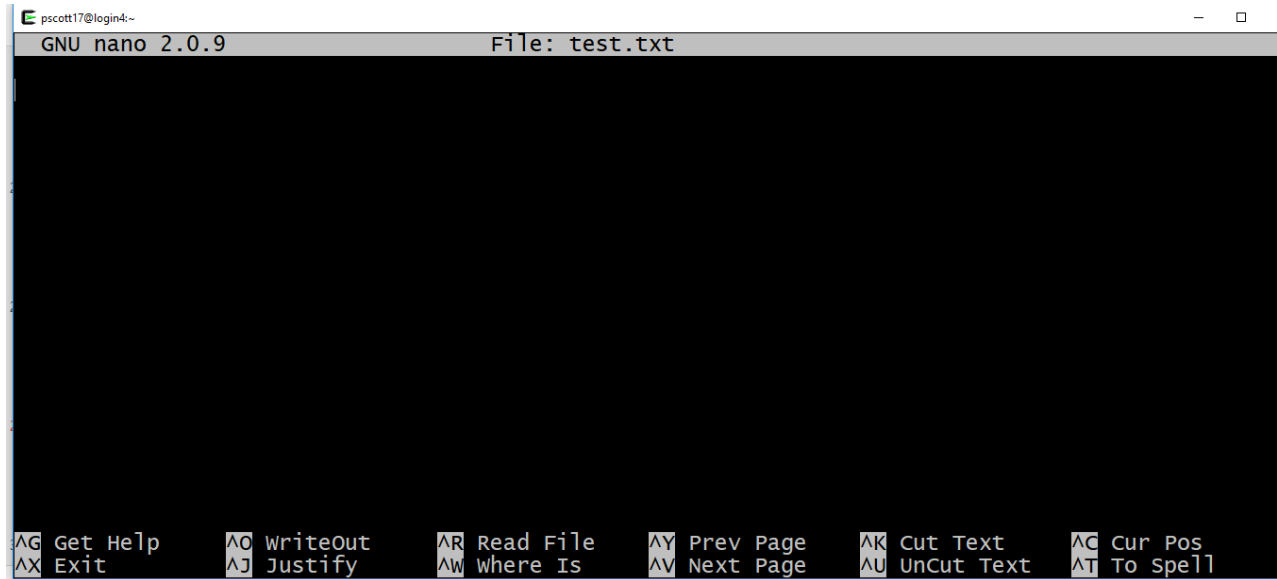
Let's practice



```
[psscott17@login3 ~]$ cd      test
[psscott17@login3 test]$ mkdir newdir
[psscott17@login3 test]$ cd newdir
psscott17@login3 newdir]$ cd ..
```

Create a text file

1. [psscott@login3 test]\$ nano test.txt



2. Type something (My first text file!).
3. Press ctrl-o (wrote out – this saves the file).
4. Press ctrl-x (exit – this will also give you options to save).
5. Lots of other text editors (vi is very popular – steeper learning curve).

Command : ls

- to list the files in the current directory

```
[psscott17@login3]$ ls
```

```
test.txt
```

```
newdir
```

Command : ls

- **ls** has many options
 - -l long list (display lots of info)
 - -s sort by modification time
 - -S sort by size
 - -h human readable
 - -r reverse order
- Options can be combined: `ls -lh`

Let's practice!



```
[pscott17@login3 test]$ ls  
newdir test.txt
```

```
pscott17@login3 test]$ ls -l  
total 8  
drwxr-xr-x 2 pscott17 hbshaffe 4096 Sep 8 09:35  
newdir  
-rw-r--r-- 1 pscott17 hbshaffe 80 Sep 8 09:50 test.txt
```

```
[pscott17@login3 test]$ ls -lh  
total 8.0K  
drwxr-xr-x 2 pscott17 hbshaffe 4.0K Sep 8 09:35  
newdir  
-rw-r--r-- 1 pscott17 hbshaffe 80 Sep 8 09:50 test.txt
```

How to know more?

- Manual
- Google

Command : man

- displays manual pages

```
[psscott17@login3 test]$ man ls
```

```
NAME
    ls - list directory contents

SYNOPSIS
    ls [OPTION]... [FILE]...

DESCRIPTION
    List information about the FILES (the current directory by default).
    Sort entries alphabetically if none of -cftuvSUX nor --sort.

    Mandatory arguments to long options are mandatory for short options
    too.

    -a, --all
        do not ignore entries starting with .
```



qto exit (windows)



ctrl-zto exit (mac)



ls sort by data



Web

Images

Shopping

Videos

News

More ▾

Search tools

About 33,700,000 results (0.77 seconds)

linux - How can I **sort** the output of 'ls' by last modified date ...

[superuser.com/.../how-can-i-sort-the-output-of-ls-by-last-modified-date](#) ▾

Apr 9, 2009 - The **ls** man page describes this in more details, and lists other options ... **ls**

-halt is for human readable , show hidden , print details , **sort** by date ...

[Linux / Unix: Sort ls Command Output By Last Modified Date ...](#)

[www.cyberciti.biz/.../ls-command-sort-the-output-by-last-modified-time-...](#) ▾

Aug 25, 2013 - Explains how to **sort** the output of **ls** command by last modified date in ...

1 vivek staff 301746331 Aug 25 01:25 **data**-db2-sample.rar -rw-r--r--@ ...

linux - How to **sort** results from **ls** command by modification ...

[unix.stackexchange.com/.../how-to-sort-results-from-ls-c...](#) ▾ Stack Exchange ▾

Aug 11, 2013 - **ls** -lrt. to get files and folders **sorted** by modification date, but this does

not **Sort data** in descending **order** of first column, for equal values, use ...



7 Answers

active

oldest

votes

477

```
ls -t
```

or (for reverse, most recent at bottom):

```
ls -tr
```

The [ls](#) man page describes this in more details, and lists other options.



General Syntax: *

- “*” can be used as a wildcard in Unix

```
[psscott17@login3 test]$ ls *txt  
test.txt
```

```
[psscott17@login3 test]$ ls t*t  
test.txt
```

```
[psscott17@login3 test]$ ls t*  
  
test.txt
```

Displaying a file

- Various ways to display a file in Unix
 - cat
 - less
 - head
 - tail

Command: cat

- dumps an entire file to standard output
- good for displaying short, simple files

```
[psscott17@login3 test]$ cat test.txt  
My first txt file!
```

Command: less

- Scrolling through a file without a mouse



Up and down keys
Scroll one line



space or ctrl-b
Scroll one page



qto exit (windows)



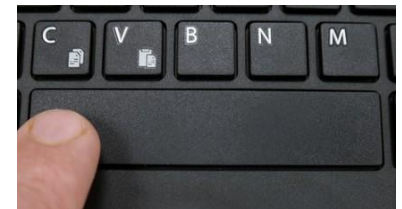
ctrl-zto exit (mac)

Let's practice!



Make a text file "large.txt" with the numbers 1-300

```
[psscott17@login3 test]$ printf '%s\n' {1..300} > large.txt
```



Let's practice!



```
[psscott17@login3 test]$ head large.txt
```

```
1
```

```
2
```

```
...
```

```
9
```

```
10
```

```
[psscot17@login3 test]$ tail large.txt
```

```
291
```

```
292
```

```
...
```

```
300
```

```
[psscott17@login3 test]$ tail -n 3 large.txt
```

```
298
```

```
299
```

```
300
```

File Commands

- Copying a file: **cp**
- Move or rename a file: **mv**
- Remove a file: **rm**
 - **There is NO going back!!!!**
 - although see:



<https://support.idre.ucla.edu/helpdesk/KB/View/6079312-i-deleted-a-file-in-my-home-directory--how-can-i-recover-it>

Copy

`cp <source> <destination>`

- To copy a file use **cp**
- **-i** (interactive)
Prompts you to confirm if the file is going to overwrite a file in your destination.
- **-r** (recursive)
 - Rather than just copying all the files and directories, copies the whole directory tree, subdirectories and all, to another location.
- **-f** (force)
 - Copies without prompting you for confirmation that the file should be overwritten.
- **-v** (verbose)
 - Will show the progress of the files being copied.

Let's practice



```
[pscott17@login3 test]$ cp test.txt text1.txt  
[pscott17@login3 test]$ ls  
large.txt newdir test1.txt test.tx test.txt
```

```
[pscott17@login3 test]$ mkdir new  
[pscott17@login3 test]$ cp -r new new2  
[pscott17@login3 test]$ ls  
large.txt new new2 newdir test1.txt test.tx test.txt
```

```
[pscott17@login3 test]$ cp test.txt new2  
[pscott17 @login3 test]$ cp test.txt new2/test_new.txt  
[pscott17@login3 test]$ cd new2  
[pscott17@login3 new]$ ls  
test_new.txt test.txt
```

Command: mv

`mv <source> <destination>`

- moves a file/directory to a different location
- renames a file/directory

```
[psscott17@login3 new2]$ cd ..  
[psscott17@login3 test]$ pwd  
/u/home/p/psscott17/test  
[psscott17@login3 test]$ mv test1.txt new  
[psscott17@login3 test]$ mv test.txt test_rename.txt  
[psscott17@login3 test]$ ls  
large.txt  new  new2  newdir  test_rename.txt  test.tx  
[psscott17@login3 test]$ mv test.tx new/test2.txt
```



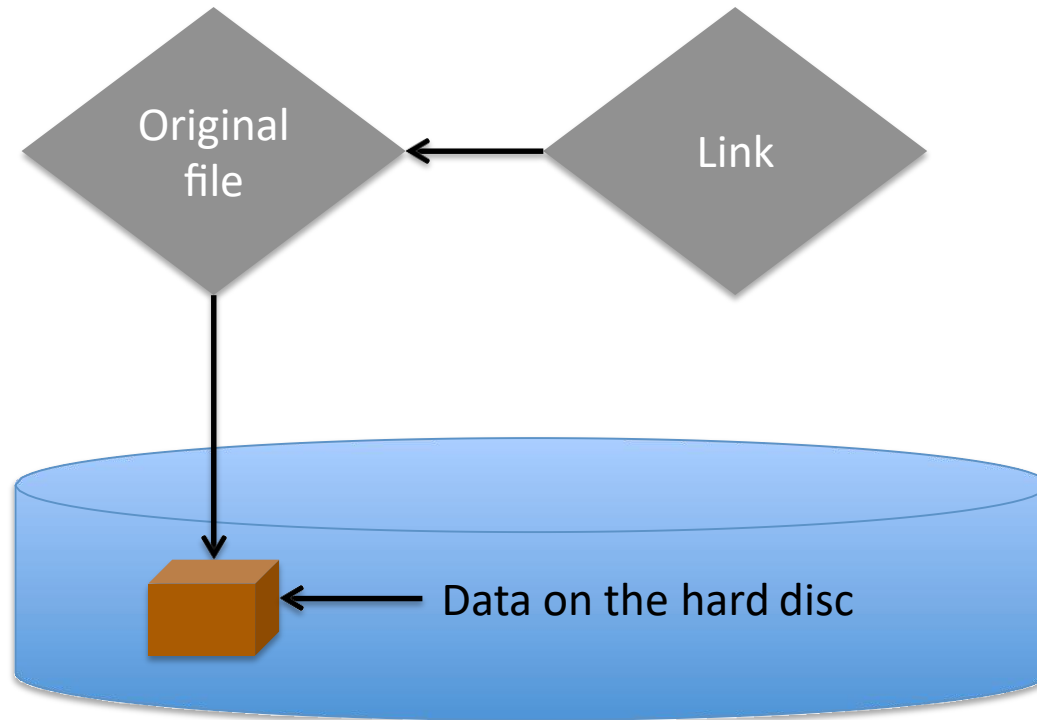
Symbolic Links

- is a special kind of file that points to another file

existing file for which you want to create the
symbolic link

name of the symbolic link

`ln <ORIGINAL_FILE> <LINK_NAME>`



Good to know



- You can perform an operation on *LINK_NAME*, just as you could with the *ORIGINAL_FILE*
- You can use normal file management commands (e.g., cp, rm) on the symbolic link.



Don't modify the original file through the link

Let's practice!



```
[psscott17@login3 new2]$ cd new
```

```
[psscott17@login3 new2]$ ln -s /u/home/p/pscot17/test/new2/ new2
```



Absolute path



New name

```
[psscott17@login3 new2]$ ls
```

```
[psscott17@login3 new2]$ less new2/test.txt
```

Command: rm

- to remove a file use **rm**
- to remove a directory use **rm -r**

```
[pscott17@login3 new]$ cd ~/test/new2  
[pscott17@login3 test]$ rm test.txt  
[pscott17@login3 test]$ cd ..  
[pscott17@login3 test]$ rm -r new2  
[pscott17@login3 test]$ ls  
large.txt  new newdir test_rename.txt
```



Files and directories deleted with **rm** are gone forever and cannot be recovered!!!

Good to know



- **cp/mv/rm** can work on many files at once:

```
cp file1 file2 new/  
rm file1 file2 file27
```

- **cp/mv/rm** can work with *:

```
mv f* new/  
rm f*  
rm l*s  
rm *txt
```

Accidental loss



- Backup your files on external hard drive
- Modify your personal Linux environment
- Remove your own write access to files you intend to not change or delete (Day 2)

Backup

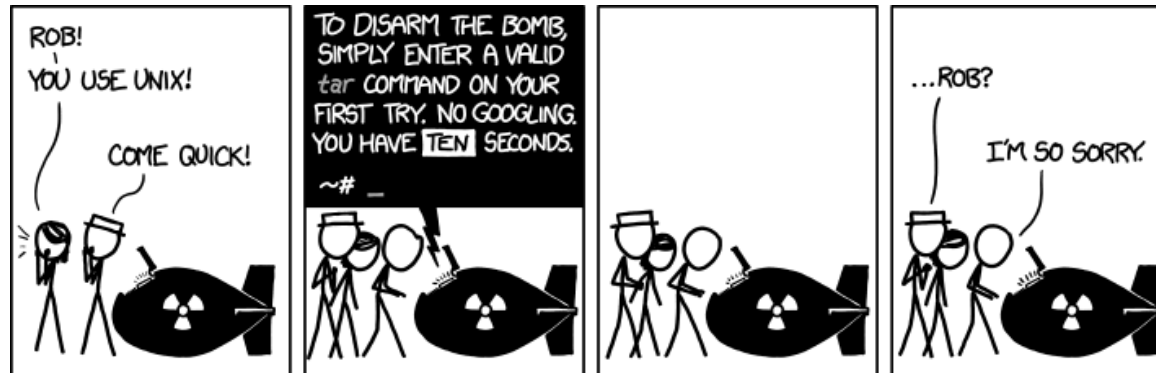
- Make backup copies of files and directories in compressed tar format
- Copy to your laptop/hard drive

```
[pscott17@login3 test]$ tar -czvf new.tgz new/
```

```
[pscott17@login3 test]$ ls -l
```

```
total 12
```

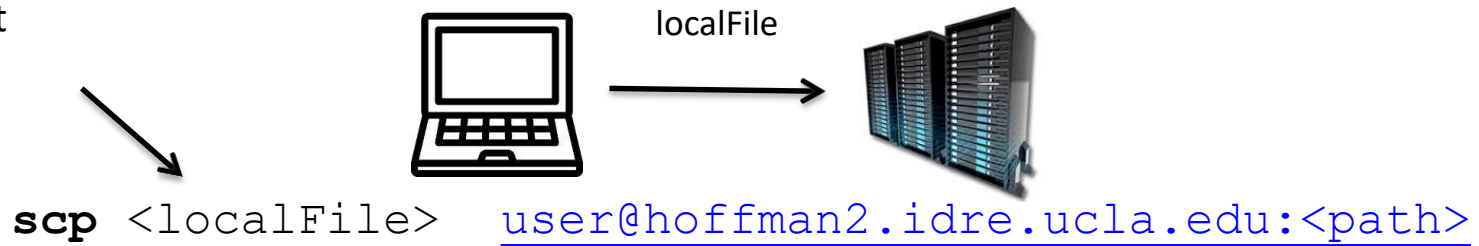
-rw-r--r--	1	pscott17	hbshaffe	255	Mar	11	10:30	large.txt
drwxr-xr-x	2	pscott17	hbshaffe	4096	Mar	11	10:55	new
-rw-r--r--	1	pscott17	hbshaffe	228	Mar	11	11:34	new.tar
-rw-r--r--	1	pscott17	hbshaffe	19	Mar	11	10:20	test_rename.txt



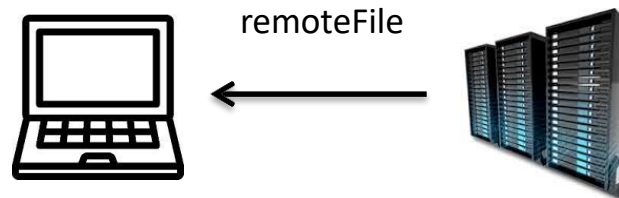
Remote copying : scp



File located on the laptop, in the current directory



Where on the cluster
<localFile> will be copied



`scp user@hoffman2.idre.ucla.edu:<path>/<remoteFile> ./`

File located on the cluster, in the <path> directory



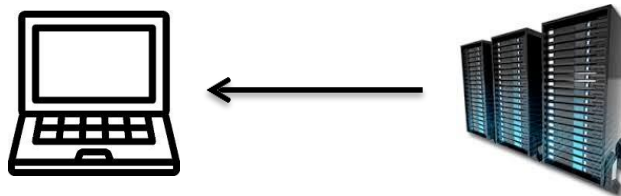
Run scp from the local session of the terminal. To open a local session :

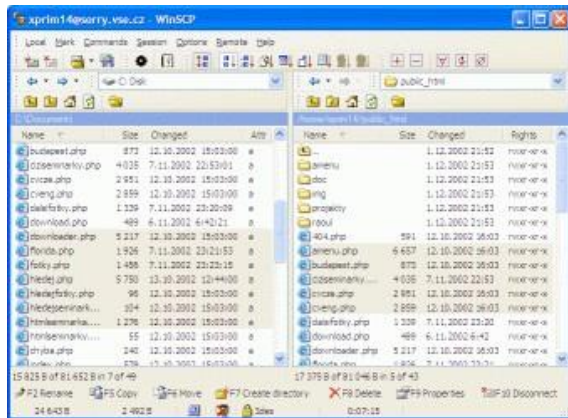
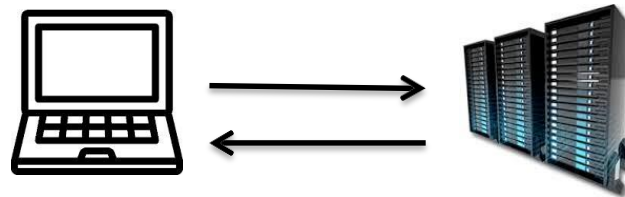
- Control-T to open a new tab
- New tab be default corresponds to a local session

Let's practice

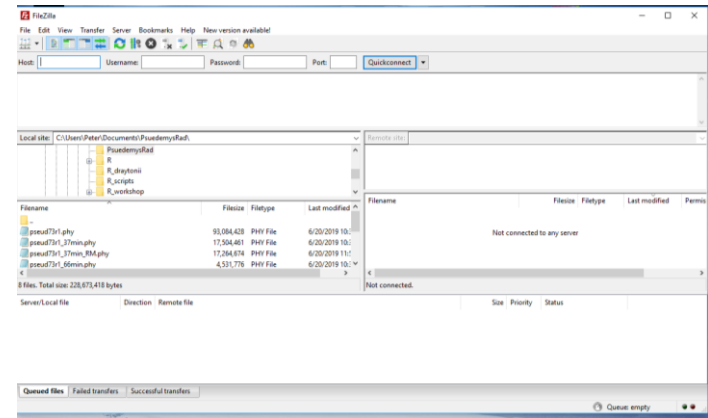


- `[users-MacBook-Air]$ scp pscott17@hoffman2.idre.ucla.edu:~/test/new.tar ./`
- `pscott17@hoffman2.idre.ucla.edu's password:`





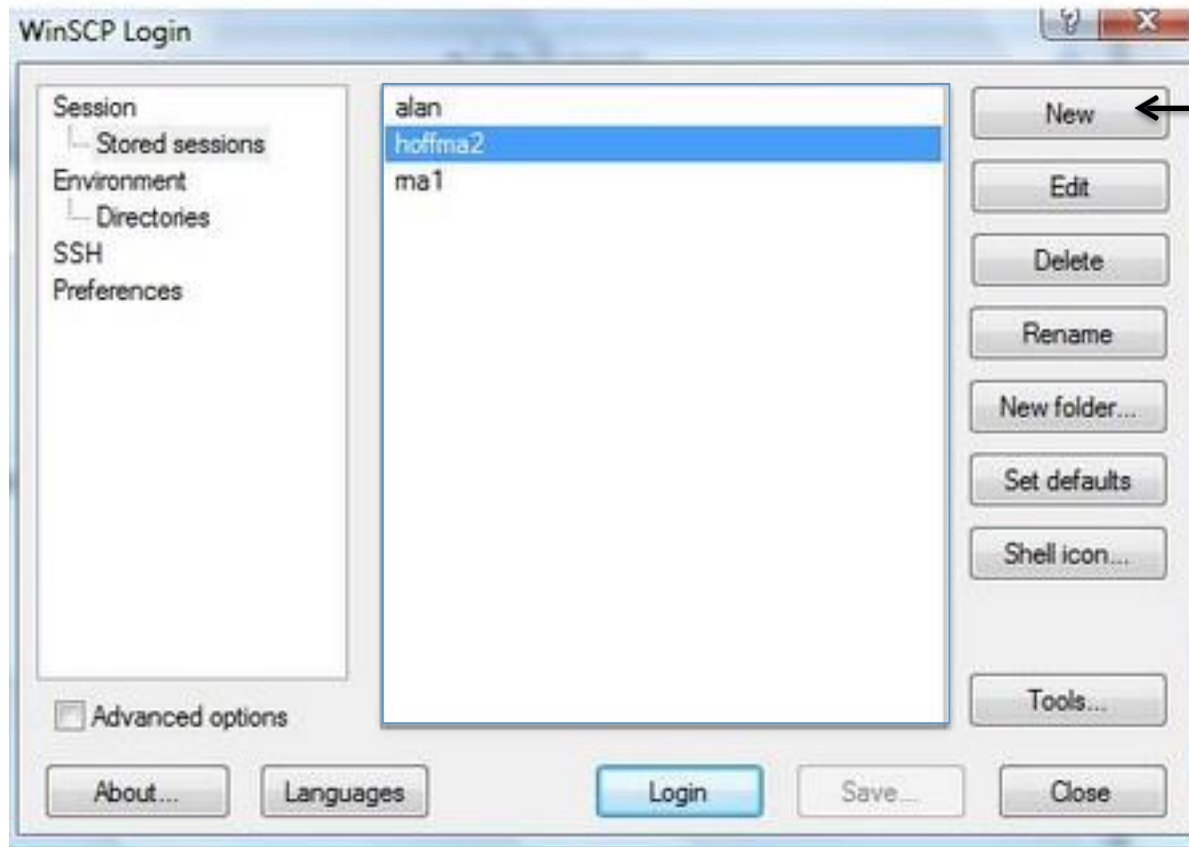
Winscp



Filezilla

Lots of Mac/Linux folks use this too!!!

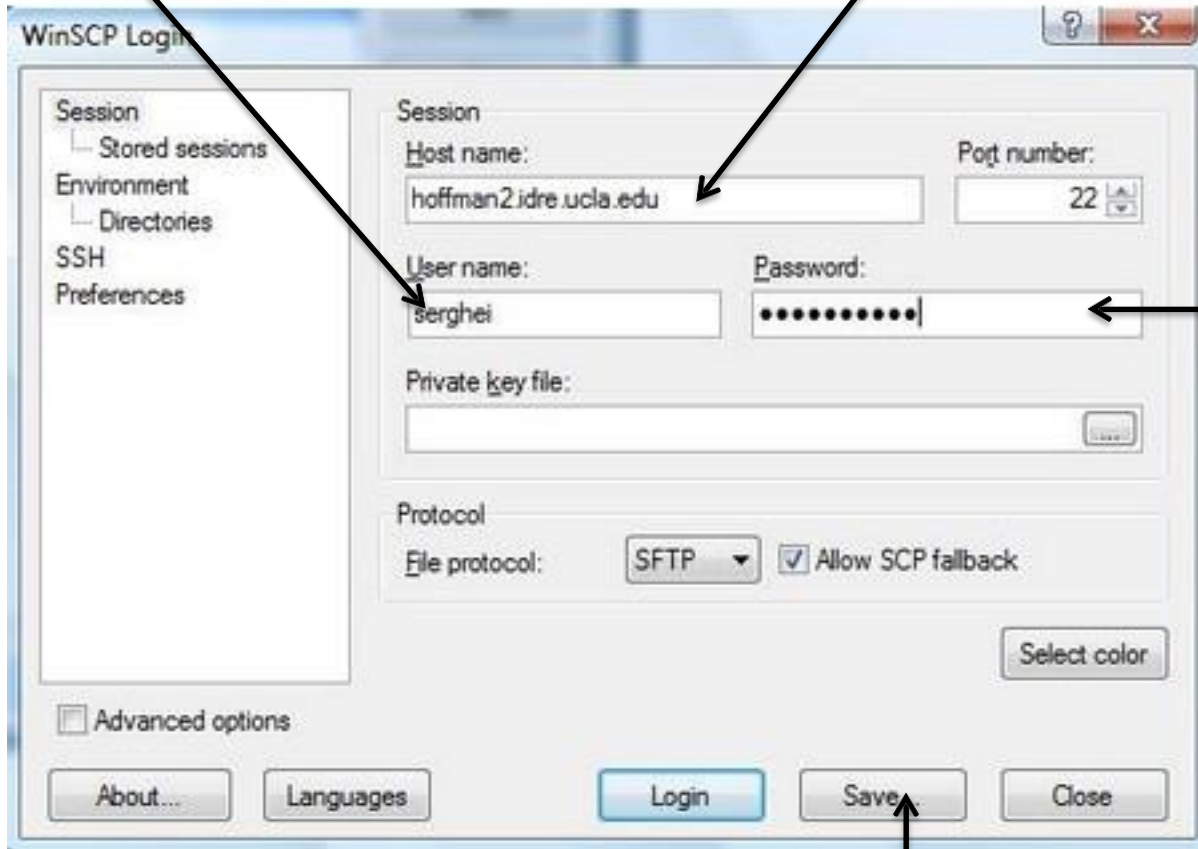
Winscp



Winscp

Username

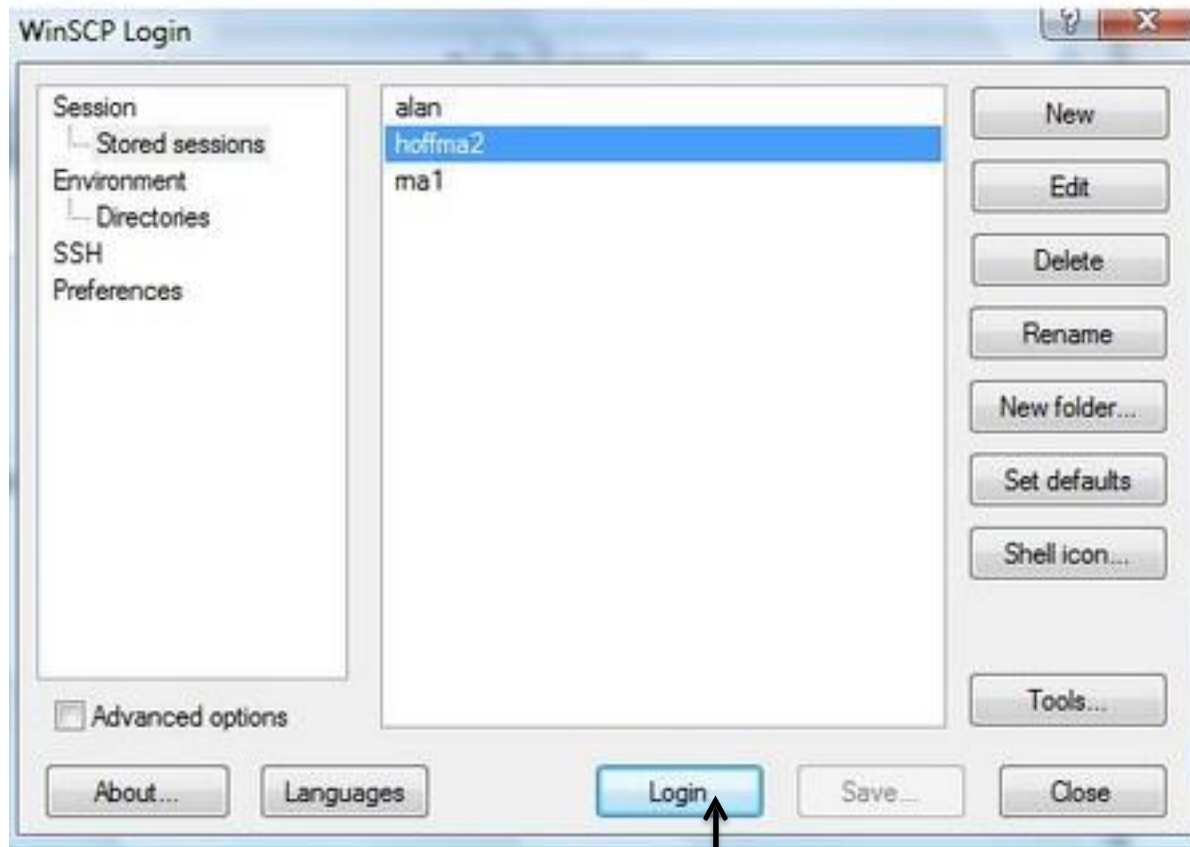
Cluster name



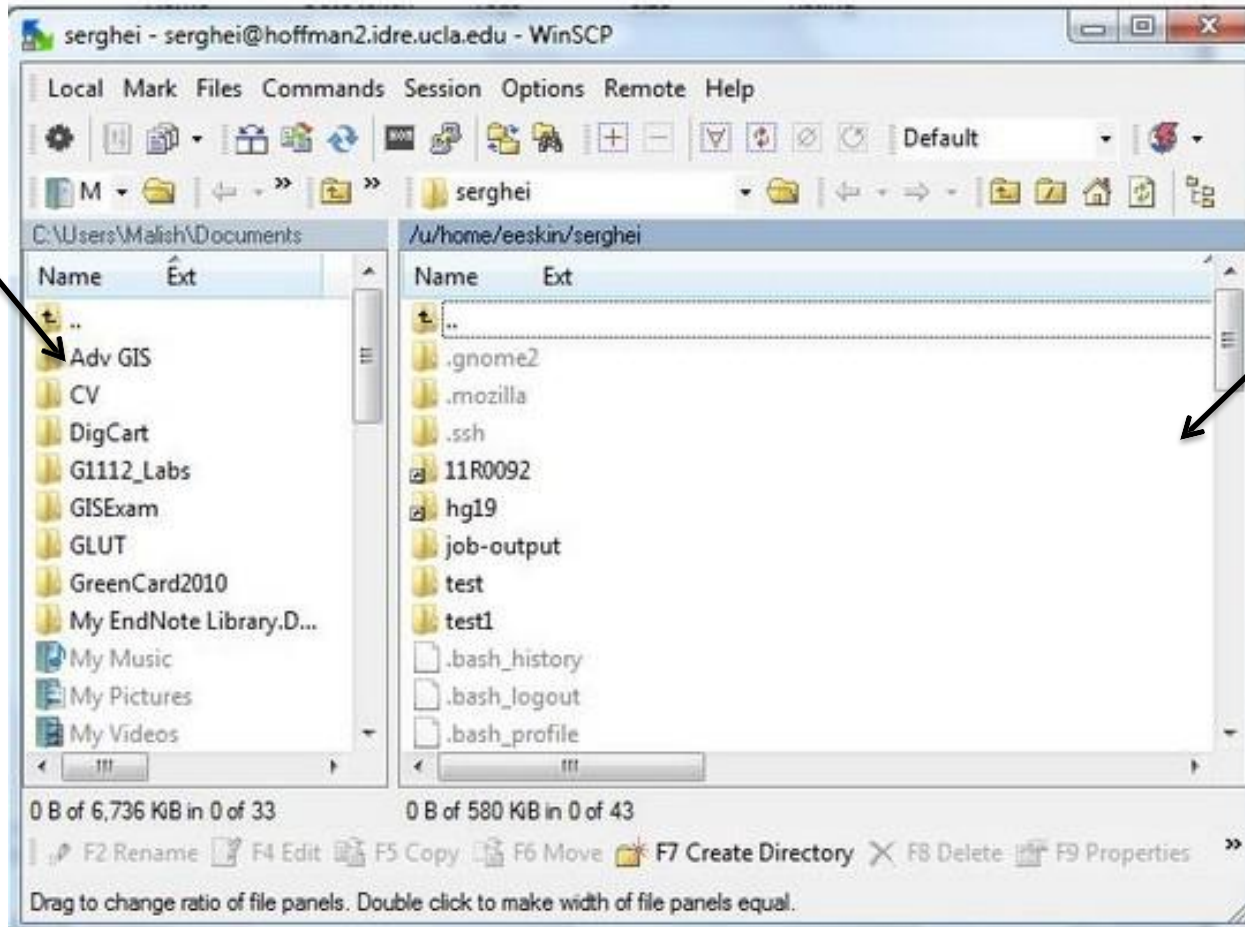
The image shows the WinSCP Login dialog box. It has a sidebar on the left with a tree view containing 'Session', 'Stored sessions', 'Environment', 'Directories', 'SSH', and 'Preferences'. The main area is titled 'Session' and contains the following fields: 'Host name:' with the value 'hoffman2.idre.ucla.edu', 'Port number:' with a spinner set to '22', 'User name:' with the value 'serghei', and 'Password:' with a masked field of dots. Below these is a 'Private key file:' field with a browse button. At the bottom, there is a 'Protocol' section with 'File protocol:' set to 'SFTP' and a checked 'Allow SCP fallback' checkbox. A 'Select color' button is to the right. At the very bottom are buttons for 'About...', 'Languages', 'Login', 'Save', and 'Close'. An arrow points from the 'Username' label to the 'User name:' field, and another from the 'Cluster name' label to the 'Host name:' field. A third arrow points from the 'Password' label to the 'Password:' field. A hand cursor icon is pointing at the 'Save' button.

Password

Winscp



Laptop



Hoffman2



To copy the files between the laptop and cluster, simply drag and drop

Modify your Linux environment

- Add prompted confirmation before any existing file is deleted or overwritten.

```
cp -i  
mv -i  
rm -i
```

Let's practice



- [psscott17@login3 test]\$ **mv -i** test_rename.txt large.txt
- mv: overwrite `test2.txt'?
- [psscott17@login3 test]\$ **rm -i** large.txt
- rm: remove regular file `test2.txt'?

Alias

- enables a replacement of a string by another string

`cp/mv/rm` \longrightarrow `cp/mv/rm -i`

- Go to home directory : `cd ~`
- Open file `.bash_profile`: `$ nano .bash_profile`
- Add in the end of the file the following lines:

```
alias cp='cp -i'
alias mv='mv -i'
alias rm='rm -i'
```

- Restart the session or `source ~/.bash_profile`

Let's practise



- [psscott17@login3 new]\$ **mv** test1.txt test2.txt
- mv: overwrite `test2.txt'?
- [psscott17@login3 new]\$ **rm** test2.txt
- rm: remove regular file `test2.txt'?

Summary

pwd -report your current directory

cd *<to where>* -change your current directory

ls *<directory>* -list contents of directory

cp *<old file>* *<new file>* -copy file

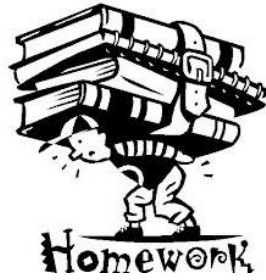
cp **-r** *<old dir>* *<new dir>* -copy a directory and its contents

mv *<old file/dir>* *<new file/dir>* -move (or rename)

rm *<file>* -delete a file

rm **-r** *<dir>* -remove a directory and its contents

mkdir *<new directory name>* -make a directory



1. Create directory “practice” in your home directory
2. Inside directory “practice” create files p.a and p.b
3. Create a copy of file p.a(p_copy.a) and rename file p.b (new name : practice.b)
4. Delete all files ending with b