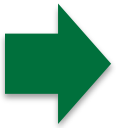
CS 50:
Software Design and Implementation

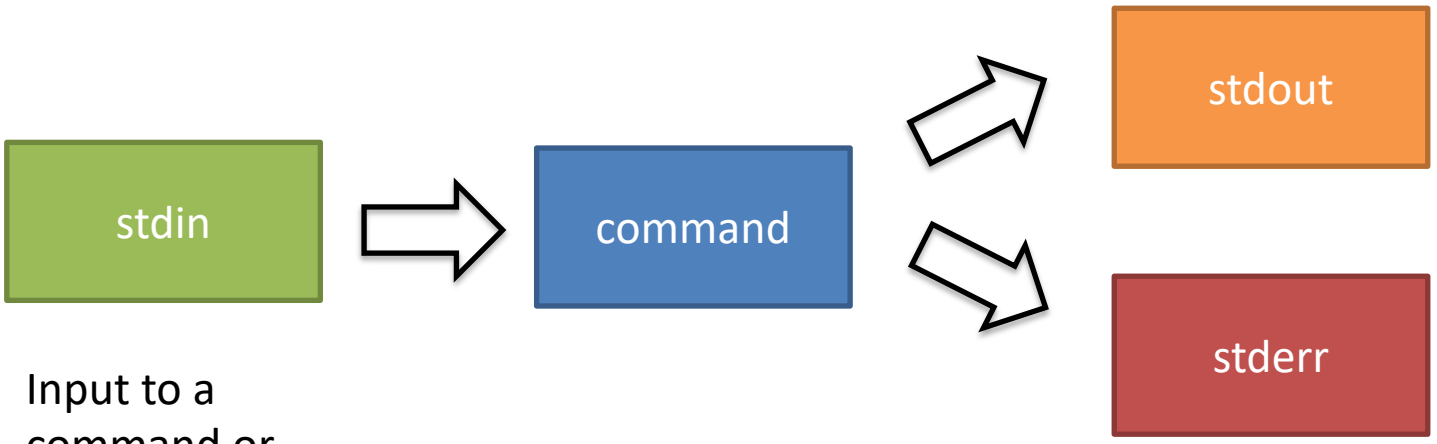
Bash Part 2

Agenda



1. Redirection
2. Pipes
3. Regular expressions and commands
4. Activity

Everything in Unix is a file, even input and output



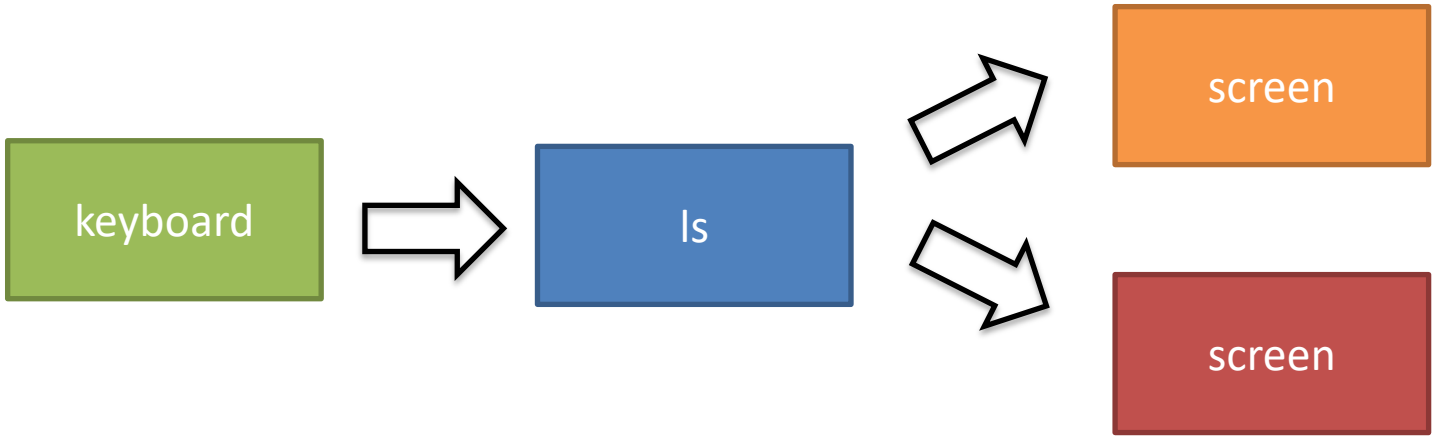
Input to a command or program is normally from stdin=keyboard

Linux treats stdin, stdout, and stderr as a file

Input and output can be redirected

Output normally goes to stdout or stderr=screen

Normally input is from the keyboard and output goes to the screen



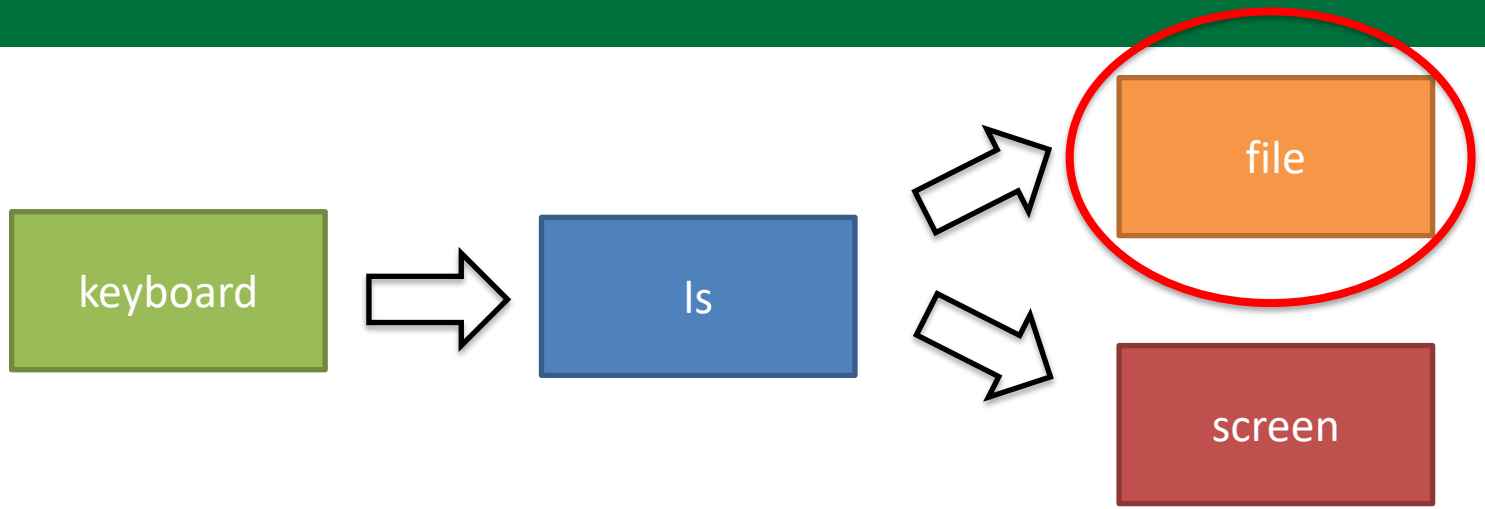

```
plank:~/cs50$ ls -l
total 8
drwxr-sr-x 3 d84xxxx thayerusers 22 Jun 26 17:04 activities/
drwxr-sr-x 3 d84xxxx thayerusers 48 Jun 20 14:22 labs/
drwxr-sr-x 3 d84xxxx thayerusers 55 Jun 26 12:27 workspace/

plank:~/cs50$ ls -l nonexistentdirectory
ls: cannot access 'nonexistentdirectory': No such file or directory
```

No errors, output goes to stdout=screen

Errors, output goes to stderr=screen

Redirect output to a file using > or >>

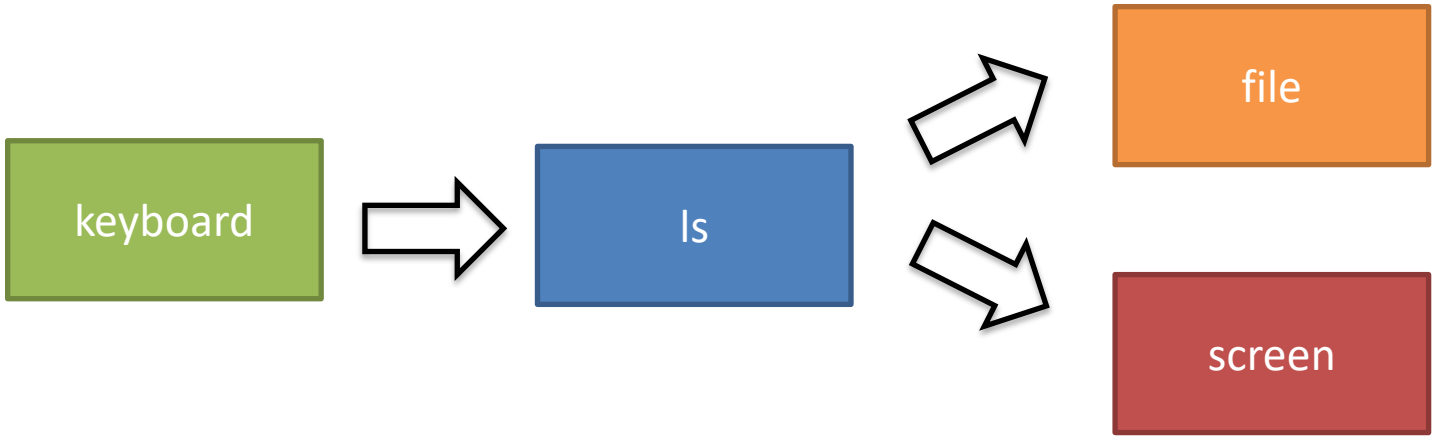



```
plank:~/cs50$ ls -l > test.txt
plank:~/cs50$ cat test.txt
total 10
drwxr-sr-x 3 d84xxxx thayerusers 22 Jun 26 17:04 activities/
drwxr-sr-x 3 d84xxxx thayerusers 48 Jun 20 14:22 labs/
-rw-r--r-- 1 d84xxxx thayerusers  0 Jun 26 18:07 test.txt
drwxr-sr-x 3 d84xxxx thayerusers 55 Jun 26 12:27 workspace/
plank:~/cs50$ ls -l nonexistentdirectory > test.txt
ls: cannot access 'nonexistentdirectory': No such file or directory
plank:~/cs50$ cat test.txt
```

Redirect output to a file (test.txt)
stdout redirected to file, no screen output

Errors, output goes to stderr=screen
Non-error output goes to stdout=file
Nothing in file (overwritten by command)

> overwrites but >> appends to file

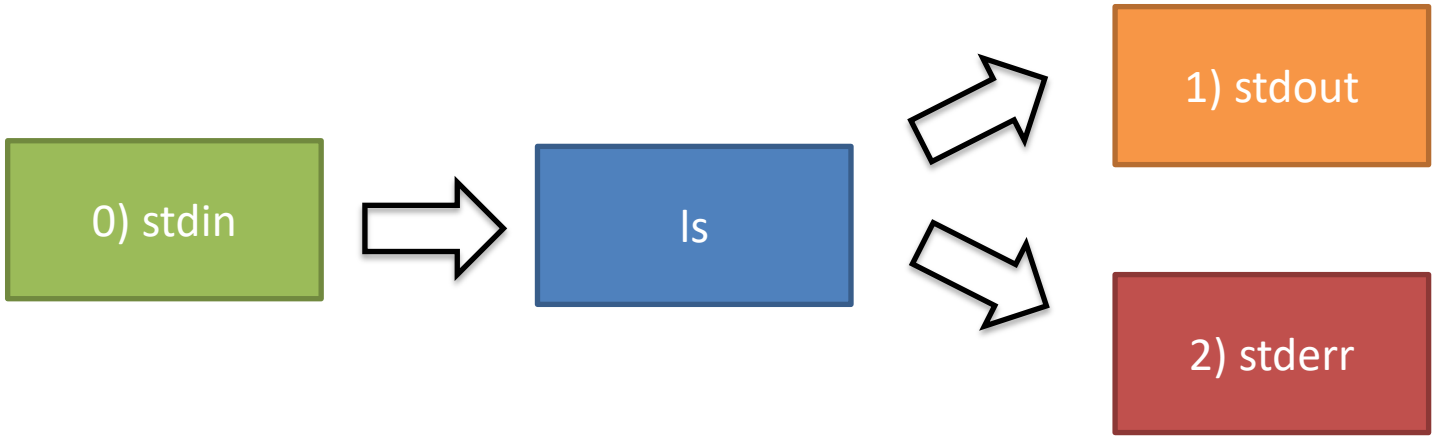



```
plank:~/cs50$ ls -l >> test.txt
plank:~/cs50$ ls -l >> test.txt
plank:~/cs50$ cat test.txt
total 10
drwxr-sr-x 3 d84xxxx thayerusers 22 Jun 26 17:04 activities/
drwxr-sr-x 3 d84xxxx thayerusers 48 Jun 20 14:22 labs/
-rw-r--r-- 1 d84xxxx thayerusers  0 Jun 26 18:08 test.txt
drwxr-sr-x 3 d84xxxx thayerusers 55 Jun 26 12:27 workspace/
total 10
drwxr-sr-x 3 d84xxxx thayerusers  22 Jun 26 17:04 activities/
drwxr-sr-x 3 d84xxxx thayerusers  48 Jun 20 14:22 labs/
-rw-r--r-- 1 d84xxxx thayerusers 243 Jun 26 18:14 test.txt
drwxr-sr-x 3 d84xxxx thayerusers  55 Jun 26 12:27 workspace/
```

Use >> to append
Run command twice

Output appears twice (second run is appended to first)

Redirect stdout and stderr with 1 and 2



To send stdout and stderr to same place use 2>$1

```
plank:~/cs50$ ls labs/ workspace/
labs/:
lab1/  test.txt

workspace/:
students/  vaccine.csv
plank:~/cs50$ ls labs/ nonexist/ 1>output.txt 2>error.txt
lank:~/cs50$ cat output.txt
labs/:
lab1/
test.txt
plank:~/cs50$ cat error.txt
ls: cannot access 'nonexist/': No such file or directory
```

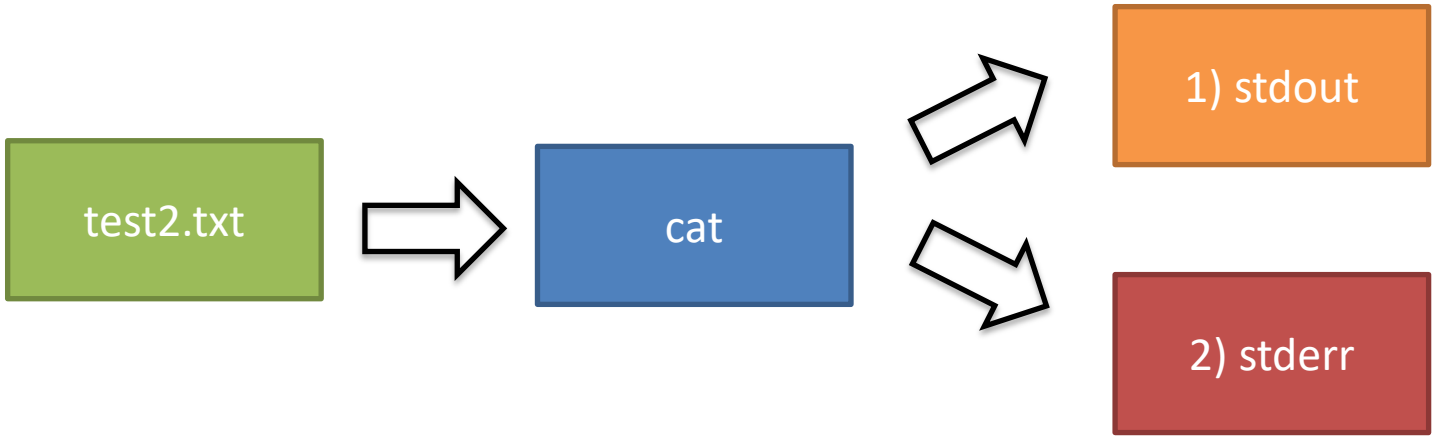
ls command can list multiple directories
Here both exist and output not redirected

Here labs exists, but nonexist does not
stdout redirected to output.txt
stderr redirected to error.txt

Non-error output to output.txt

Error output to error.txt

Redirect stdin with <



To send stdout and stderr to same place use 2>$1

```
plank:~/cs50$ cat > test2.txt
This is a test
This is only a test
plank:~/cs50$ cat < test2.txt
This is a test
This is only a test
```

If file name not given to cat, uses stdin for input
Here we redirect cat output to test2.txt

Typed from keyboard, use ctrl-d to signal end of file
cat output redirected to test2.txt

< redirects input from test2.txt instead of keyboard
(could have just typed cat test2.txt)

Agenda

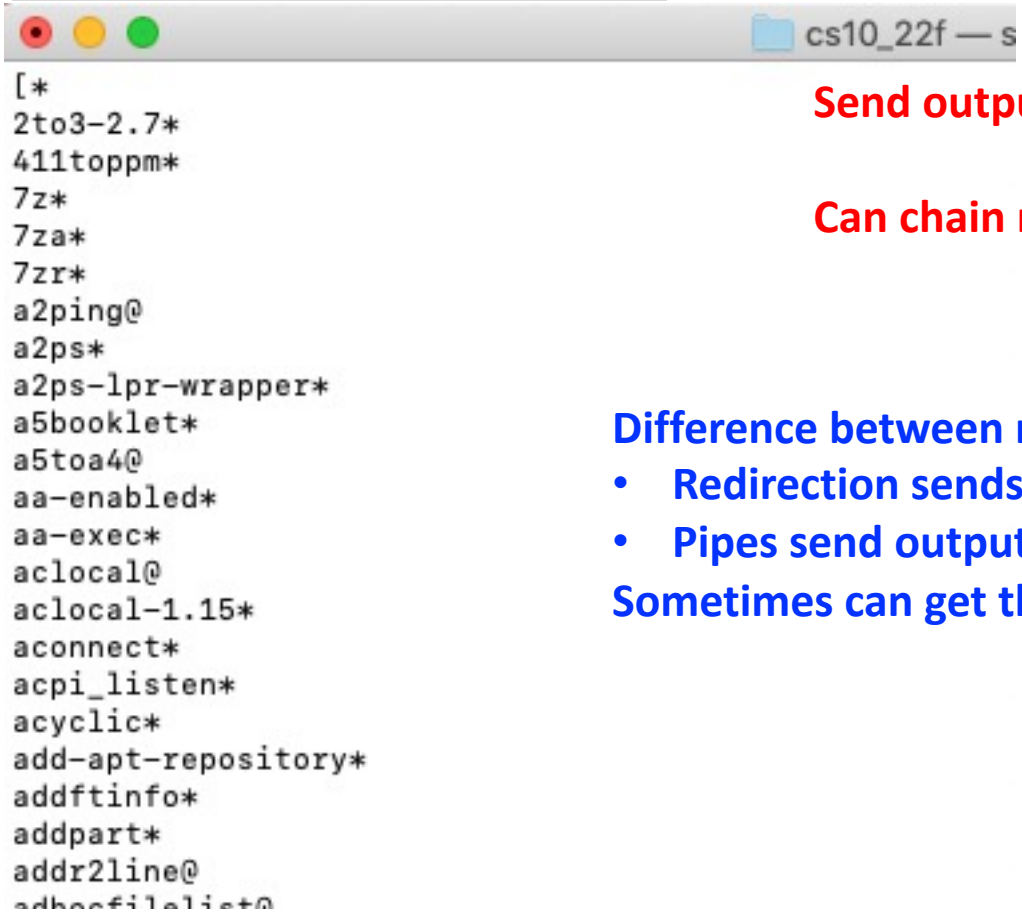
1. Redirection
2. Pipes
3. Regular expressions and commands
4. Activity

Redirection sends output to file; pipes send output from one command to another

Pipe format

command | command | command

plank:~/cs50$ **ls /usr/bin | less**



Many files are in /usr/bin (location of binaries for commands)

Send output of ls to less using pipe |

Can chain many commands together into a pipeline

Difference between redirection and pipes:

- **Redirection sends output to file**
- **Pipes send output to next command**

Sometimes can get the same result using either

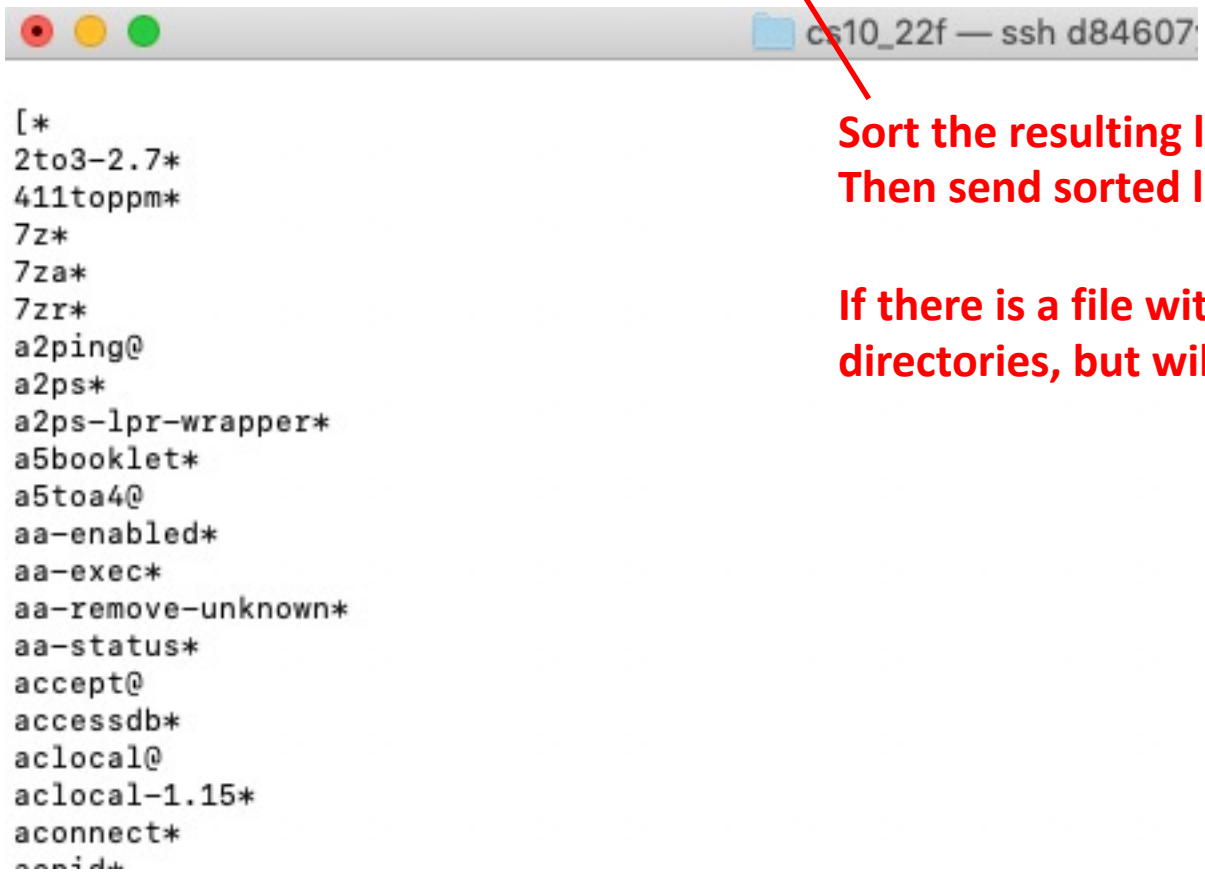
Commands can be linked together into a pipeline

Pipe format

command | command | command

List multiple directories

```
plank:~/cs50$ ls /usr/bin /usr/sbin | sort | less
```



Sort the resulting list of both directories
Then send sorted list to less

If there is a file with the same name in both directories, but will appear twice in less

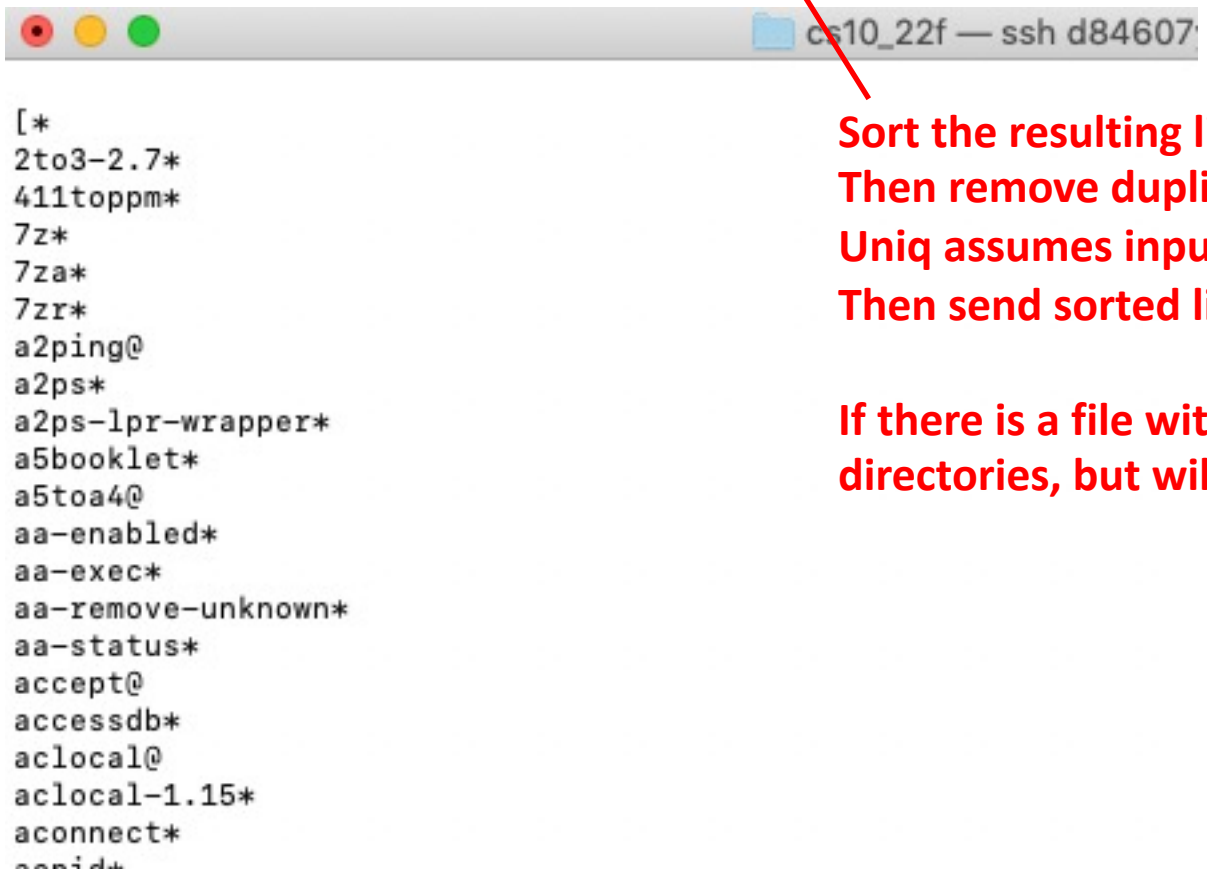
Add uniq to remove duplicates

Pipe format

command | command | command

List multiple directories

```
plank:~/cs50$ ls /usr/bin /usr/sbin | sort | uniq | less
```



Sort the resulting list of both directories
Then remove duplicates with uniq
Uniq assumes input sorted
Then send sorted list to less

If there is a file with the same name in both directories, but will appear once in less

Translate input using tr

```
plank:~/cs50$ echo "lower case letters" | tr a-z A-Z
LOWER CASE LETTERS
```

Echo sends text to tr command
Tr translates letters a-z to capital A-Z

Adapted from: The Linux Command Line by William Shotts, no starch press, 2019

The famous ROT13 cipher is easily implemented (and broken) with tr

```
plank:~/cs50$ echo "lower case letters" | tr a-z A-Z
LOWER CASE LETTERS
plank:~/cs50$ echo "A secret message!" | tr a-zA-Z n-za-mN-ZA-M
N frperg zrffntr!
plank:~/cs50$ echo "N frperg zrffntr!" | tr a-zA-Z n-za-mN-ZA-M
A secret message!
```

Echo sends text to tr command
Tr translates letters a-z to capital A-Z

ROT13 is a special form of the Cesear cipher

ROT13 is a substitution cipher where each letter is rotated 13 characters to the right to encode

To decode, rotate 13 characters to the left

Notice: ! character is not encoded. Why?

Adapted from: *The Linux Command Line* by William Shotts, no starch press, 2019

Agenda

1. Redirection
2. Pipes
3. Regular expressions and commands
4. Activity

Regular expressions identify patterns in text

Regular expression special characters

grep [options] regex [file…]

plank:~/cs50$ **grep -i 'grafton' vaccine.csv**
//list of all lines in vaccine.csv with the word grafton in it
//case insensitive with –i flag

plank:~/cs50$ **grep '^03/28/2022' vaccine.csv**
//list of all lines that start with 03/28/22

Crossword puzzle:
Find all five letter words that have third letter j and last letter r
Linux has a dictionary at /usr/share/dict/american-english

grep -i '^..j.r$' /usr/share/dict/american-english

Char	Meaning
.	Match a single character
*	Match any number of characters
^	Start of line
$	End of line
()	Group

Use \ to escape if looking for special characters (e.g., \.)

Adapted from: *The Linux Command Line* by William Shotts, no starch press, 2019

Use cut command to read a column from a file

```
plank:~/cs50$ head passwd
root:x:0:0:Root:/root:/bin/bash
timroot:x:0:0:Tim Tregubov Root:/root:/bin/bash
wbcroot:x:0:0:Wayne as Root:/root:/bin/bash
mattroot:x:0:0:Matt as Root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown

plank:~/cs50$ cut -d : -f 1 passwd | head
root
timroot
wbcroot
mattroot
bin
daemon
adm
lp
sync
shutdown
```

File to search

-f specifies first column
(1 indexed)

-d specifies delimiter between fields
Here delimiter is :
Use ‘ ‘ for space

sed selects lines from a file that match a regular expression

```
@plank:cs50$ cd $loc/workspace/activities/day3
@plank:day3$ cat people.txt
Homer A. Simpson
Marge B. Simpson
Bart C. Simpson
Lisa D. Simpson
Maggie F. Simpson
Montomery G. Burns
Sideshow H. Bob
Millhouse I. Van Houten
Ned J. Flanders
Edna K. Krabappel
Barney L. Gumble
Kustry T. Clown

plank:day3cs50$ sort people.txt | sed –n '/Edna/, /Lisa/p'
Edna K. Krabappel
Homer A. Simpson
Kustry T. Clown
Lisa D. Simpson
```

See what is in file

Find entries from Edna to List
-n flag says don't print all lines
(otherwise does)
/p means print lines that match

Sort

sed selects lines from a file that match a regular expression

```
plank:day3$ sort people.txt | sed -n '/Simpson$/p'
Bart C. Simpson
Homer A. Simpson
Lisa D. Simpson
Maggie F. Simpson
Marge B. Simpson

plank:day3$ sort people.txt | sed -n -e '/Maggie/d' -e '/Simpson/p'
Bart C. Simpson
Homer A. Simpson
Lisa D. Simpson
Marge B. Simpson
plank:~/cs50$ cat people.txt
Homer A. Simpson
Marge B. Simpson
Bart C. Simpson
Lisa D. Simpson
Maggie F. Simpson
Montomery G. Burns
Sideshow H. Bob
Millhouse I. Van Houten
Ned J. Flanders
Edna K. Krabappel
Barney L. Gumble
Kustry T. Clown
```

Find only Simpsons
Note: $ means ends with

-e means multiple commands
/d means delete
Here we delete Maggie and print the remaining Simpsons

Maggie is not deleted from original file, only from stream of text return from sed command

sed selects lines from a file that match a regular expression

```
plank:~/cs50$ sort people.txt | sed 's/Simpson/Pierson/' | cat -n
     1 Barney L. Gumble
     2 Bart C. Pierson
     3 Edna K. Krabappel
     4 Homer A. Pierson
     5 Kustry T. Clown
     6 Lisa D. Pierson
     7 Maggie F. Pierson
     8 Marge B. Pierson
     9 Millhouse I. Van Houten
    10 Montomery G. Burns
    11 Ned J. Flanders
    12 Sideshow H. Bob
```

Simpsons enter witness protection and get a new last name!

s/<find>/<replace>/

Add line numbers to output with –n flag

Agenda

1. Redirection
2. Pipes
3. Regular expressions and commands
4. Activity

Summary of commands

Command	Name	Use
>	Redirect	Redirect stdout and overwrite `ls –l > output.txt`
>>	Redirect	Redirect stdout and append `ls –l >> output.txt`
<	Redirect	Redirect stdin cat < text.txt
|	Pipe	Send output of command to next command `sort text.text | uniq`
sort	sort	Sort input `sort text3.txt`
uniq	unique	Remove duplicates (expects prior sort) `sort test3.txt | uniq`
tr	Translate	Translate input to output `tr a-z A-Z < test3.txt`
cut	cut	Return a column (from –f) from a file using delimiter (-d) `cut -d : -f 1`
sed	Stream editor	Search and replace text in a file (among other capabilities) sed –n 's/Simpson/Pierson/p'