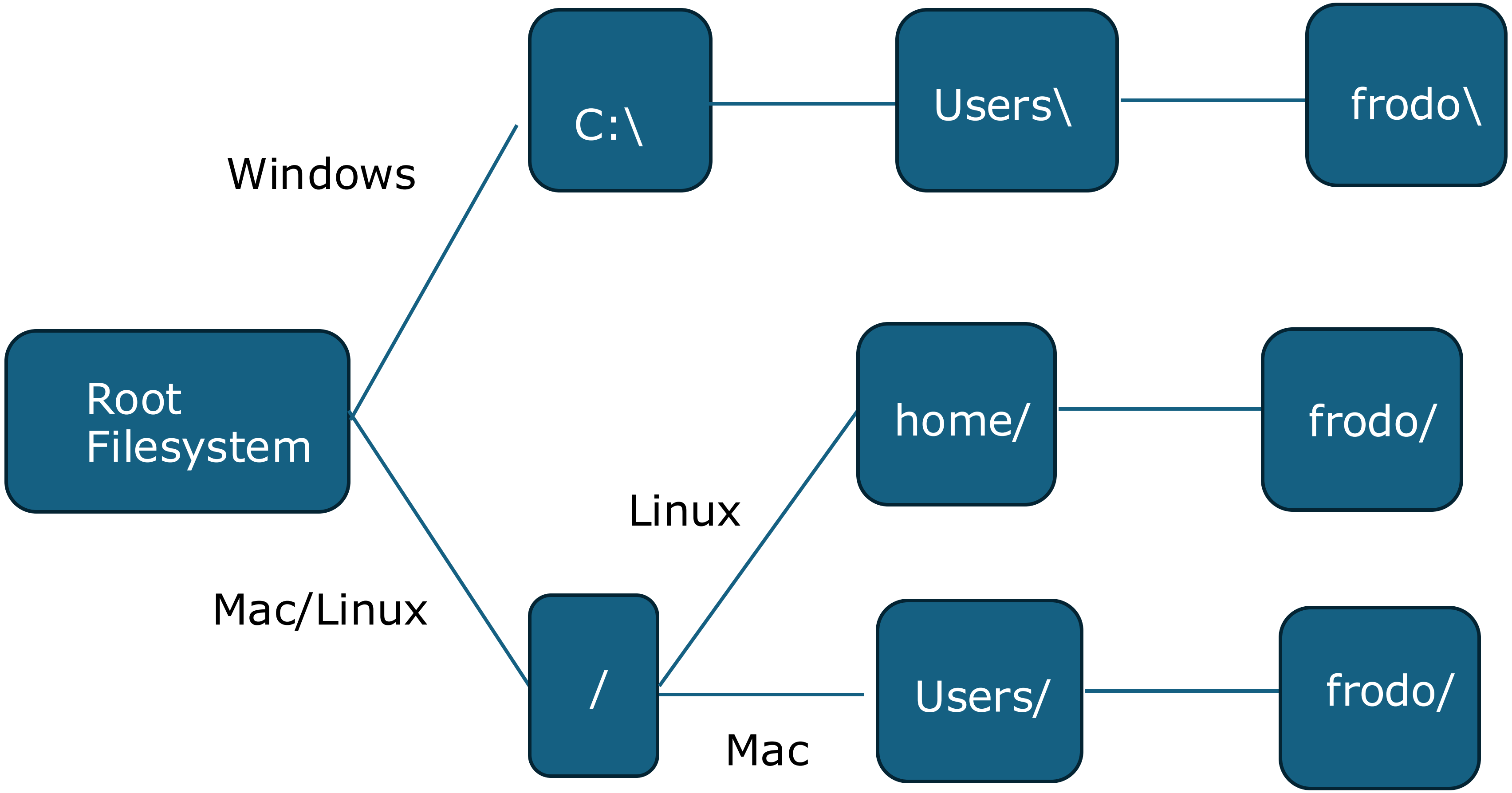
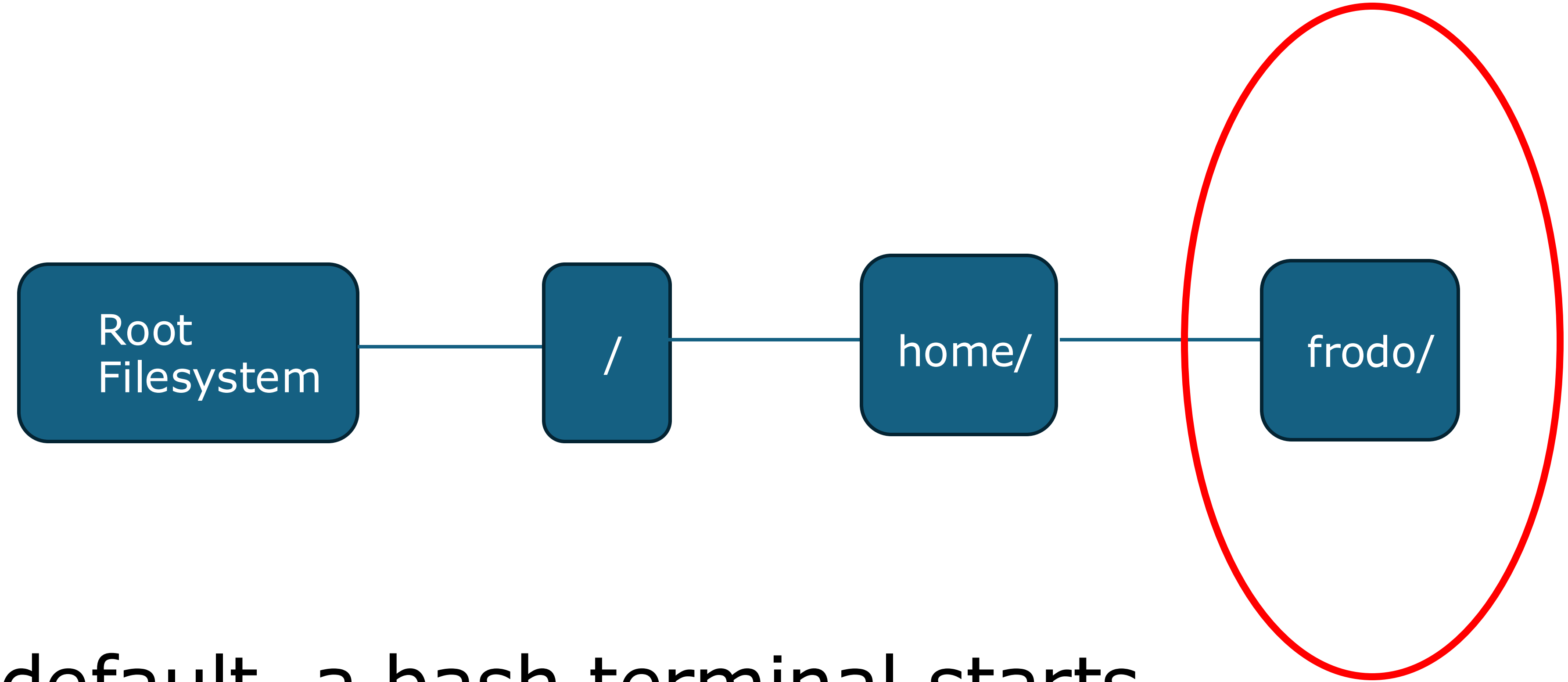
The Bash Command Line for Chemists

The Computer Filesystem – getting to a user’s (frodo) home directory:

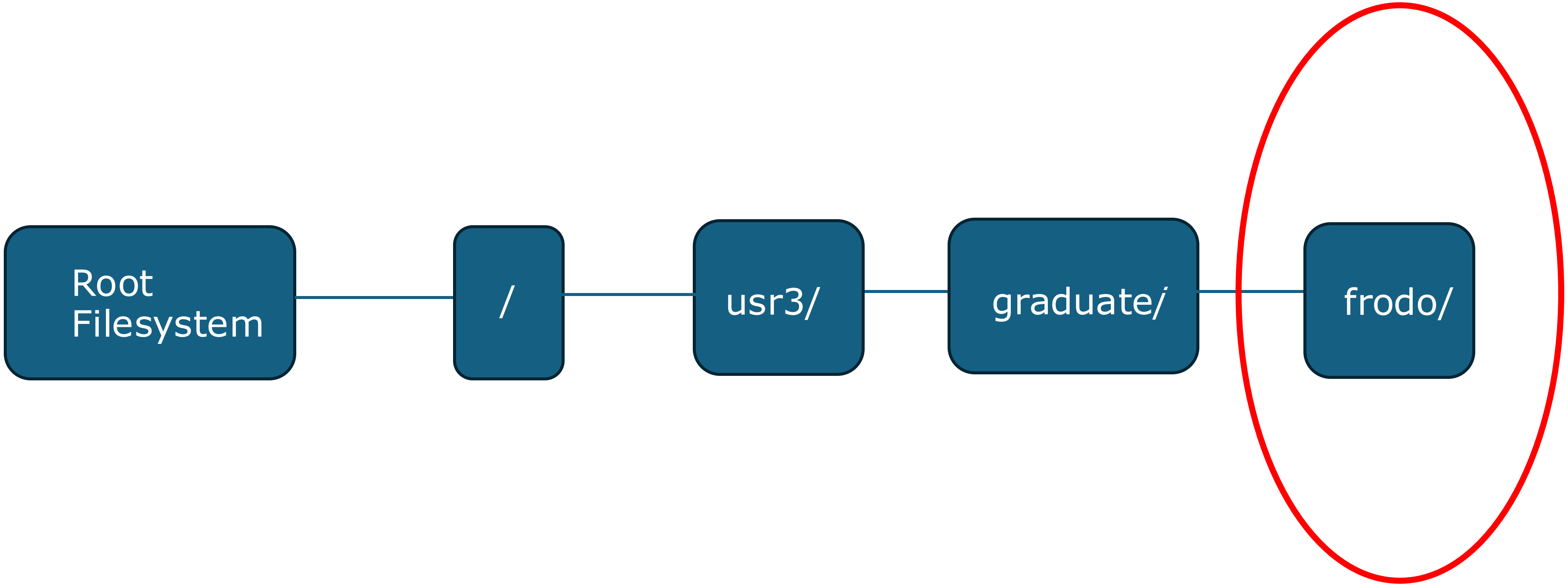


The bash terminal “exists” in a certain location on a filesytem



By default, a bash terminal starts in a user’s home directory

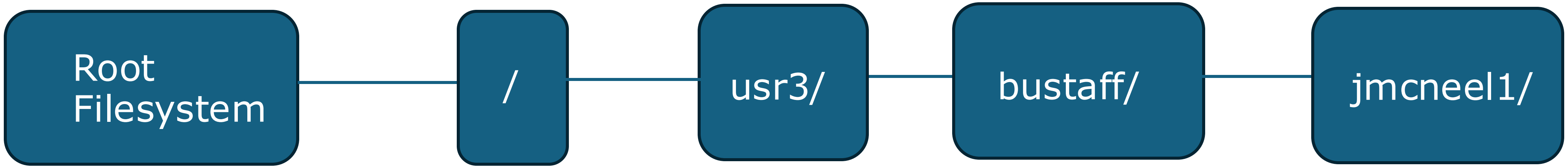
The bash terminal “exists” in a certain location on a filesytem

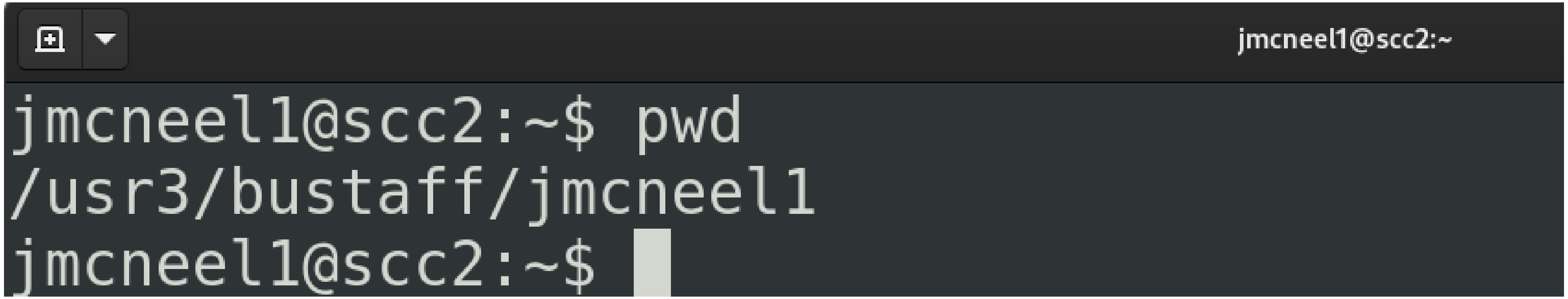


By default, on the SCC a bash terminal starts in a user’s home directory

The first command you’ll learn: pwd

- pwd: **P**resent **W**orking **D**irectory

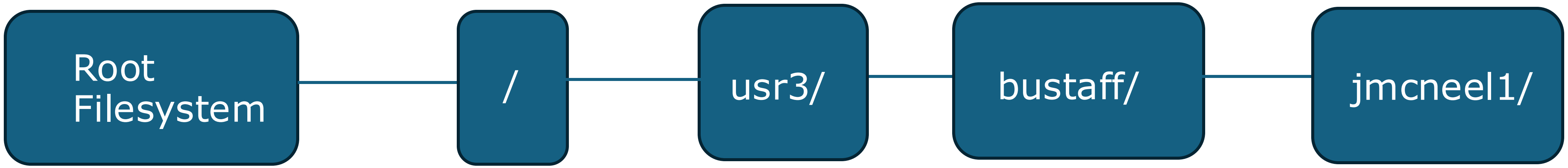


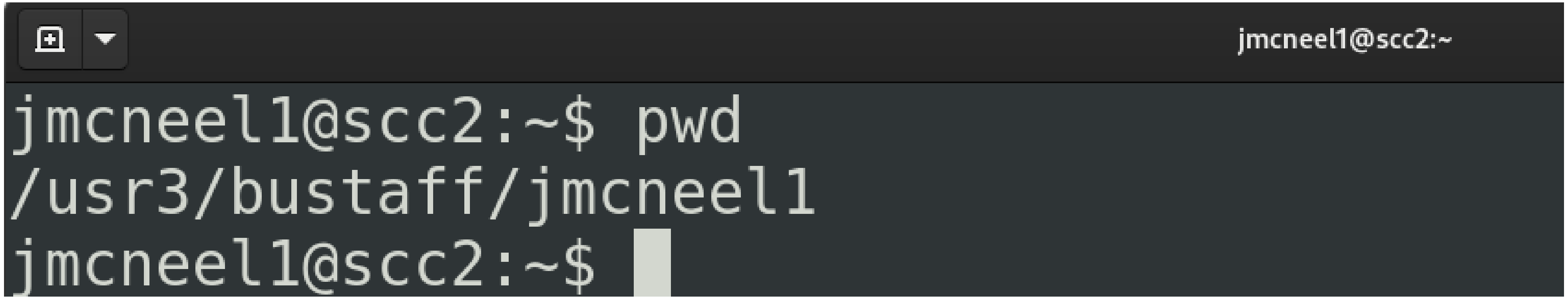


```
jmcneel1@scc2:~$ pwd
/usr3/bustaff/jmcneel1
jmcneel1@scc2:~$ 
```

The first awful rhyme of the day:

- The terminal can only see what’s in your PWD.

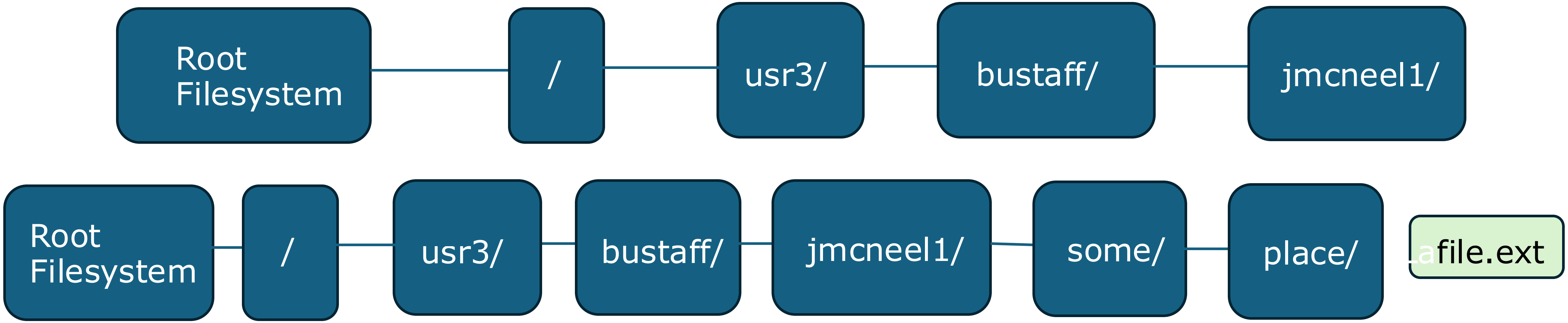




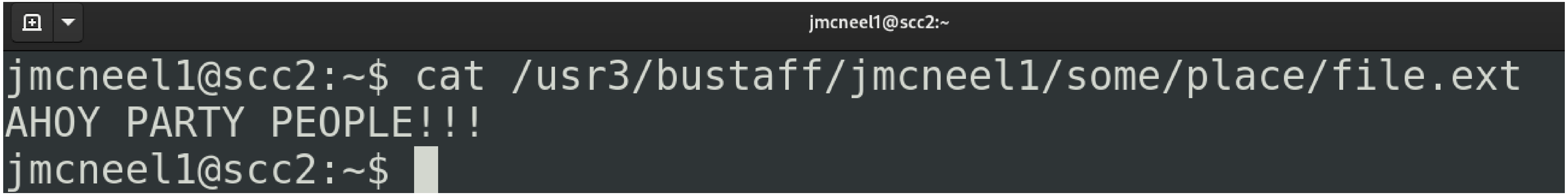
```
jmcneel1@scc2:~$ pwd
/usr3/bustaff/jmcneel1
jmcneel1@scc2:~$ 
```

How do you interact with files in places other than your PWD?

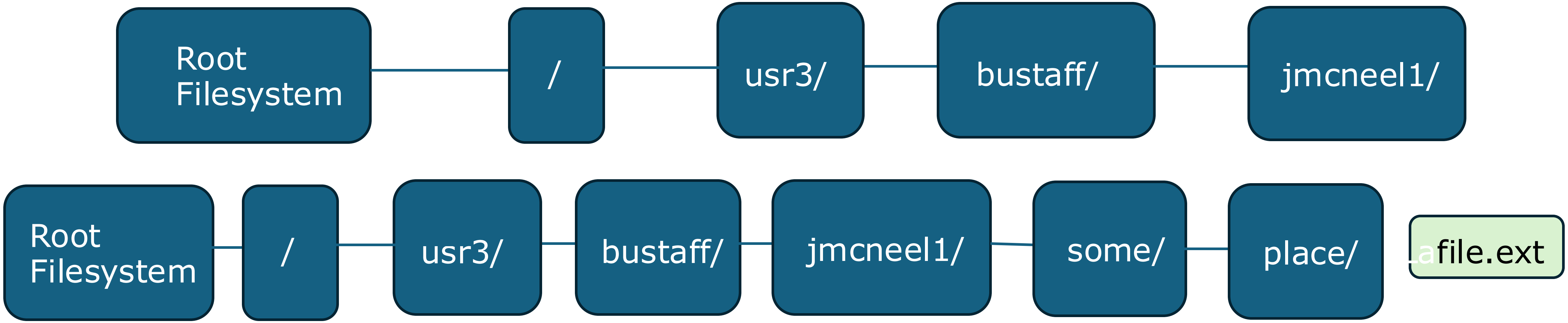
1. Use an absolute file path.
2. Use a relative file path.
3. Move your PWD to the location of the file.



Use an absolute file path



```
jmcneel1@scc2:~$ cat /usr3/bustaff/jmcneel1/some/place/file.ext
AHOY PARTY PEOPLE!!!
jmcneel1@scc2:~$ 
```

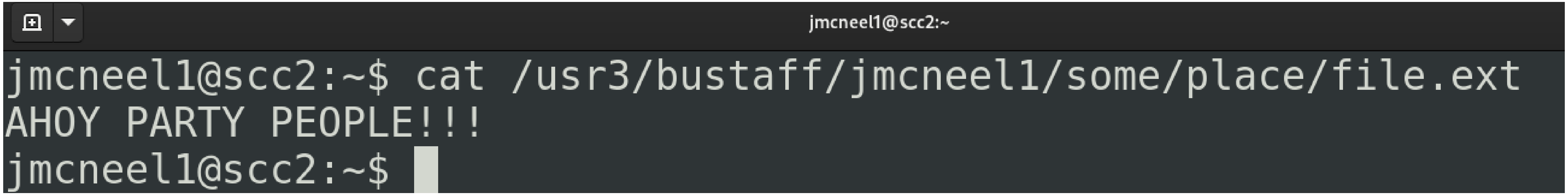


Here we used a new command: cat

- cat: con**CAT**enate
- command prints the contents of a file

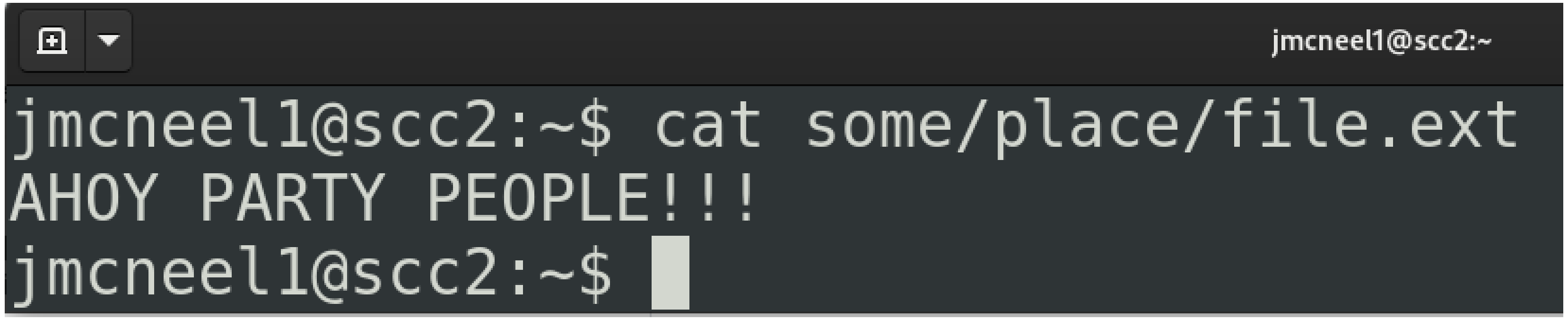
Rhyme Number Two Y’All!

To print files just like that, hit the file with cat

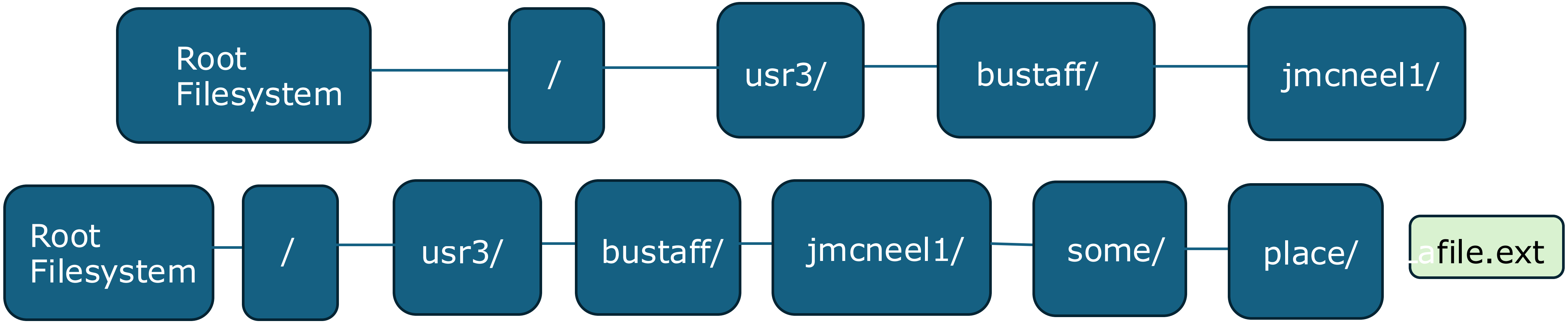


```
jmcneel1@scc2:~$ cat /usr3/bustaff/jmcneel1/some/place/file.ext
AHOY PARTY PEOPLE!!!
jmcneel1@scc2:~$
```

Use a relative file path



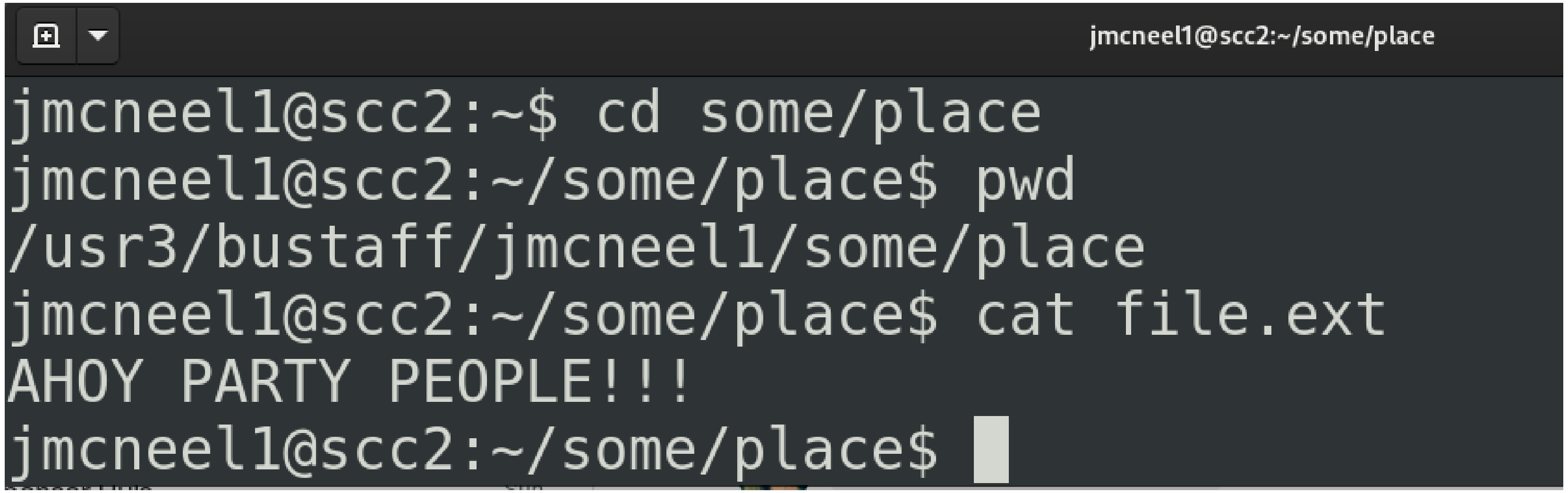
```
jmcneel1@scc2:~$ cat some/place/file.ext
AHOY PARTY PEOPLE!!!
jmcneel1@scc2:~$ 
```



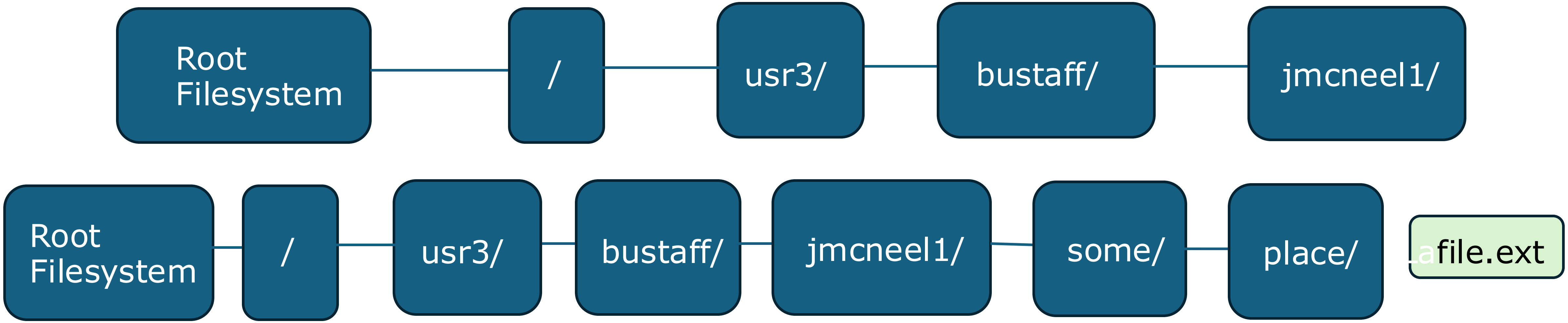
Root Filesystem — / — usr3/ — bustaff/ — jmcneel1/

Root Filesystem — / — usr3/ — bustaff/ — jmcneel1/ — some/ — place/ — file.ext

Move your pwd to the location of the file.



```
jmcneel1@scc2:~$ cd some/place
jmcneel1@scc2:~/some/place$ pwd
/usr3/bustaff/jmcneel1/some/place
jmcneel1@scc2:~/some/place$ cat file.ext
AHOY PARTY PEOPLE!!!
jmcneel1@scc2:~/some/place$ 
```



Root Filesystem — / — usr3/ — bustaff/ — jmcneel1/

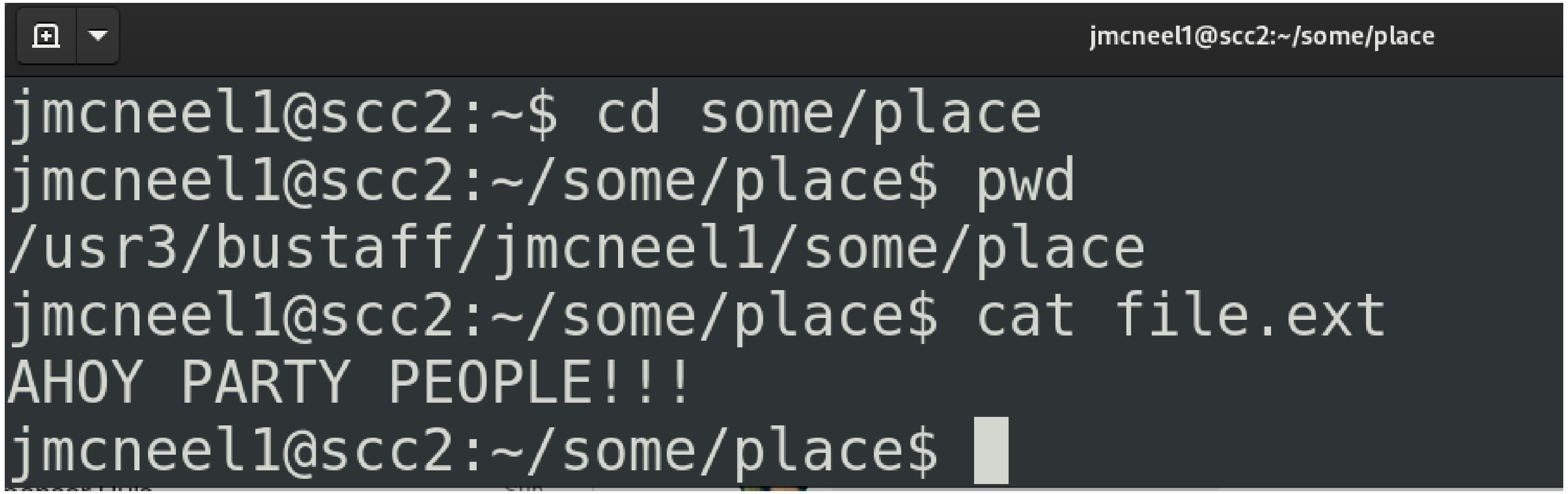
Root Filesystem — / — usr3/ — bustaff/ — jmcneel1/ — some/ — place/ file.ext

Here we used a new command: cd

- cd: **C**hange **D**irectory
- command does exactly what it says

Get Ready For Rhyme Number Three:

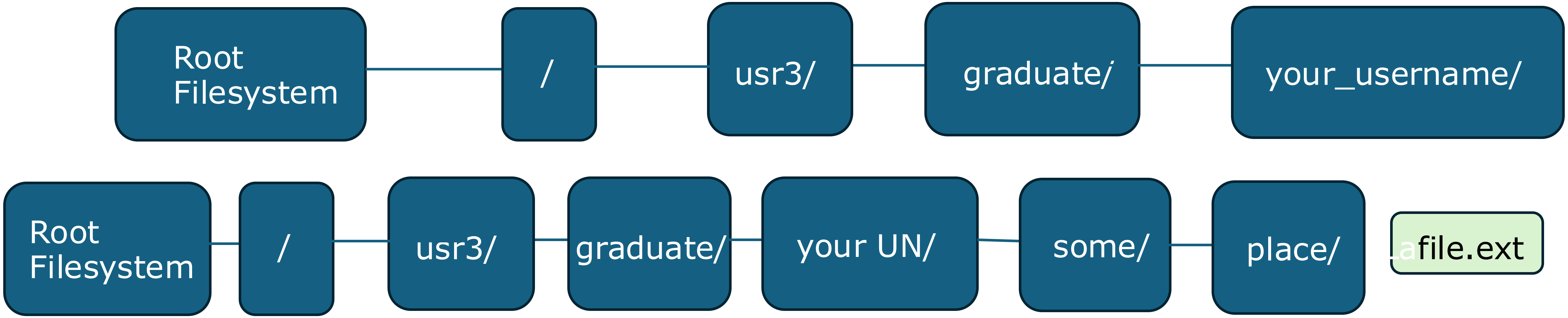
To move around the Mac or PC, use good old cd.

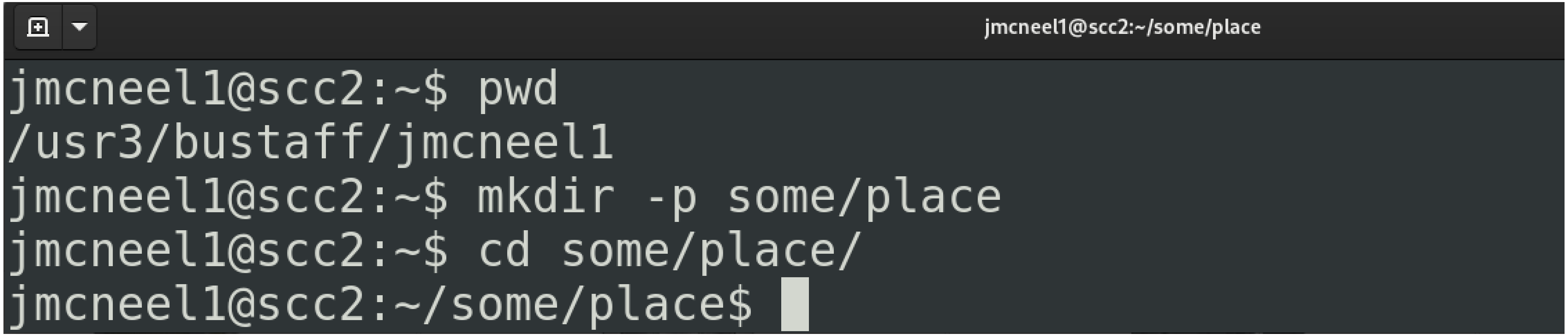


```
jmcneel1@scc2:~$ cd some/place
jmcneel1@scc2:~/some/place$ pwd
/usr3/bustaff/jmcneel1/some/place
jmcneel1@scc2:~/some/place$ cat file.ext
AHOY PARTY PEOPLE!!!
jmcneel1@scc2:~/some/place$
```

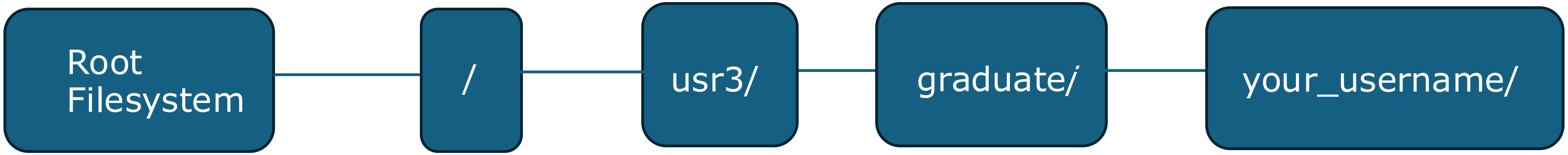
Put the ideas into practice.

- Let’ s create the directories shown below, and then move into them (with cd).
- We’ll use relative filepaths.





```
jmcneel1@scc2:~$ pwd
/usr3/bustaff/jmcneel1
jmcneel1@scc2:~$ mkdir -p some/place
jmcneel1@scc2:~$ cd some/place/
jmcneel1@scc2:~/some/place$ 
```



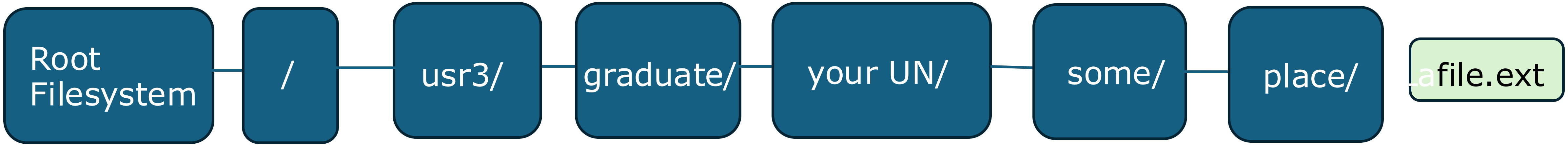
Root Filesystem

/

usr3/

graduate/

your_username/



Root Filesystem

/

usr3/

graduate/

your UN/

some/

place/

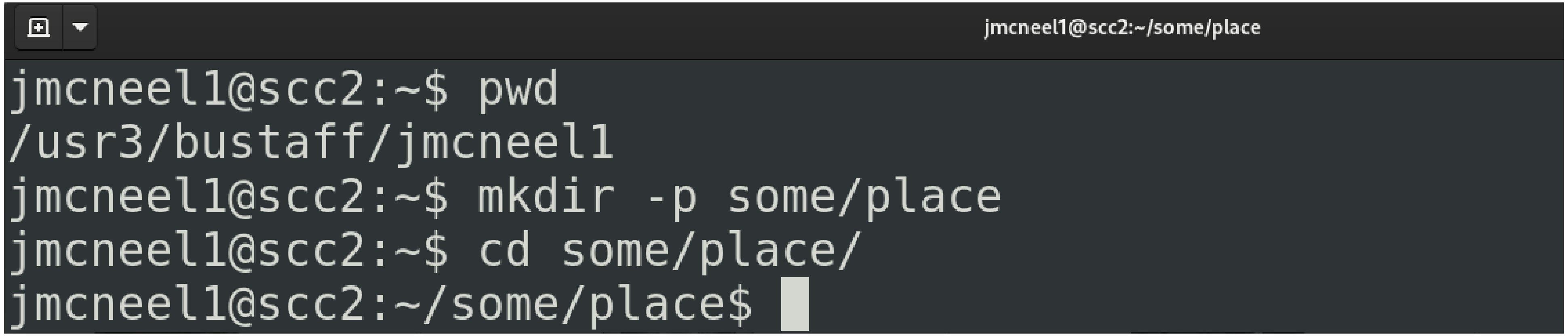
file.ext

A new command!! mkdir

- mkdir: **M**a**K**e **DIR**ectory
- command does exactly what it says
- -p is a flag that tells the computers to mkdir recursively.

Rhyme Number Four is upon us:

To create folders and organize your files as you prefer, use the great util mkdir.



```
jmcneel1@scc2:~$ pwd
/usr3/bustaff/jmcneel1
jmcneel1@scc2:~$ mkdir -p some/place
jmcneel1@scc2:~$ cd some/place/
jmcneel1@scc2:~/some/place$ 
```

The “back” shortcut

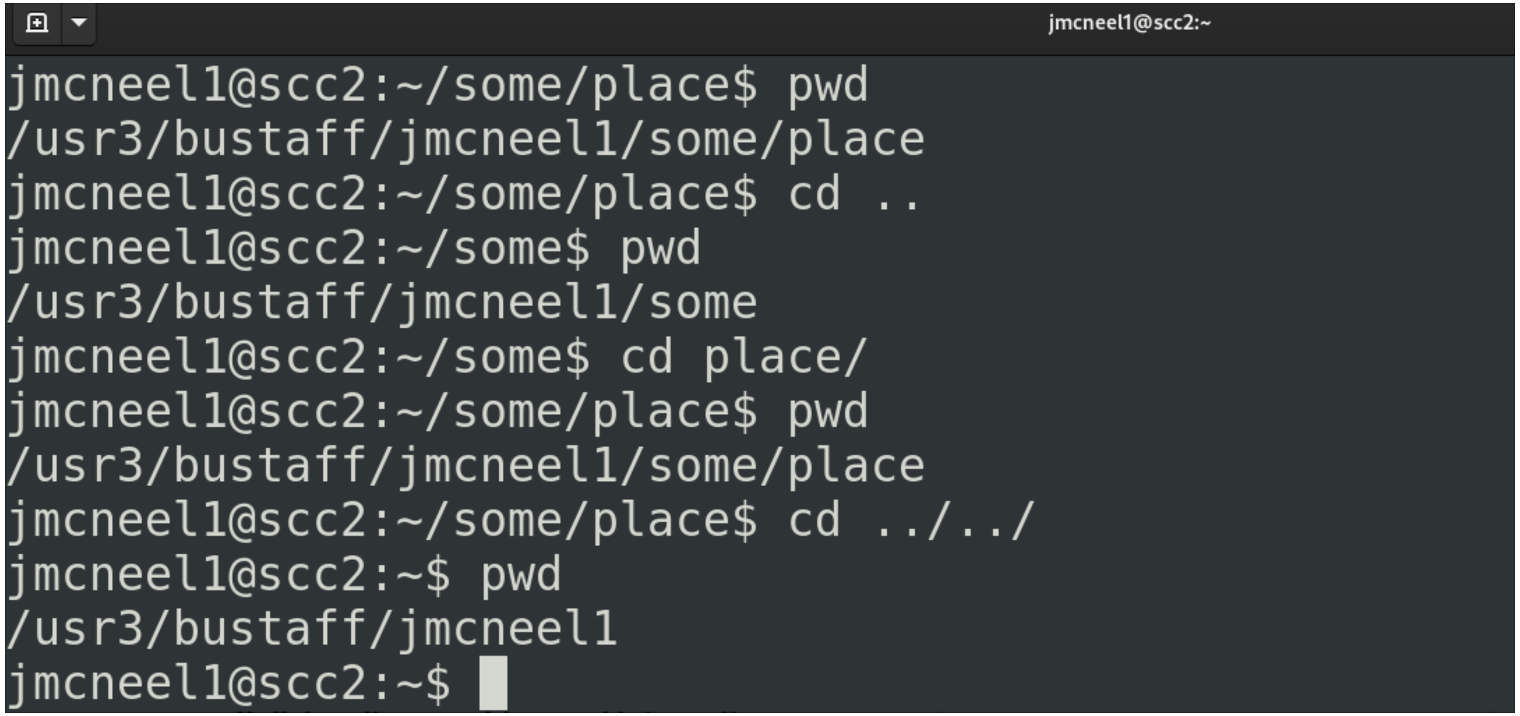
As we’ve already shown, you can always use an absolute filepath the move around your filesystem

Using relative paths, however, is often MUCH faster

But how do you move “back” a directory, i.e. going from /usr3/graduate/UN/some/place to /usr3/graduate/UN/some?

Use ..

Use ..



```
jmcneel1@scc2:~/some/place$ pwd
/usr3/bustaff/jmcneel1/some/place
jmcneel1@scc2:~/some/place$ cd ..
jmcneel1@scc2:~/some$ pwd
/usr3/bustaff/jmcneel1/some
jmcneel1@scc2:~/some$ cd place/
jmcneel1@scc2:~/some/place$ pwd
/usr3/bustaff/jmcneel1/some/place
jmcneel1@scc2:~/some/place$ cd ../../
jmcneel1@scc2:~$ pwd
/usr3/bustaff/jmcneel1
jmcneel1@scc2:~$
```

Shortcuts to get around the filesystem

The Home Directory: ~

~ = /usr3/graduate/YourUserName

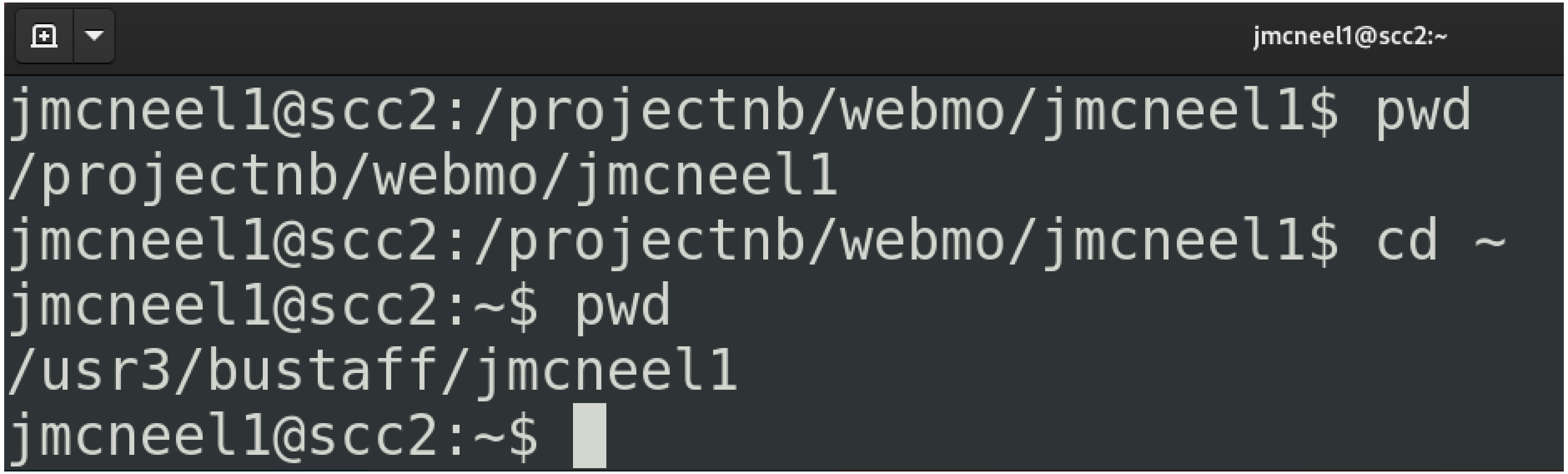
Tabbing Around:

Use the tab key to autofill directory and filenames

Shortcuts to get around the filesystem

The Home Directory: ~

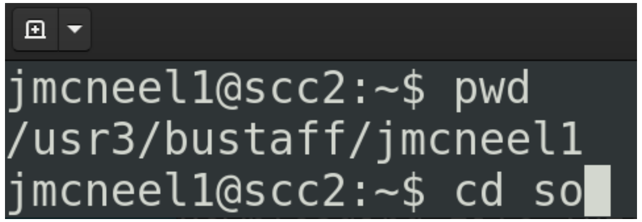
~ = /usr3/graduate/YourUserName



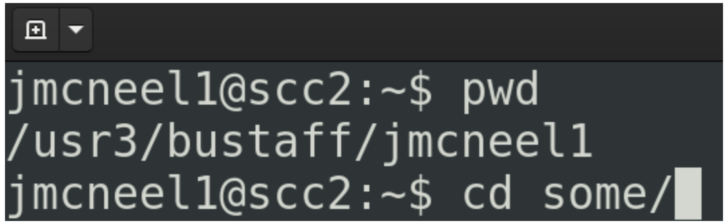
```
jmcneel1@scc2:/projectnb/webmo/jmcneel1$ pwd
/projectnb/webmo/jmcneel1
jmcneel1@scc2:/projectnb/webmo/jmcneel1$ cd ~
jmcneel1@scc2:~$ pwd
/usr3/bustaff/jmcneel1
jmcneel1@scc2:~$ 
```

Shortcuts to get around the filesystem

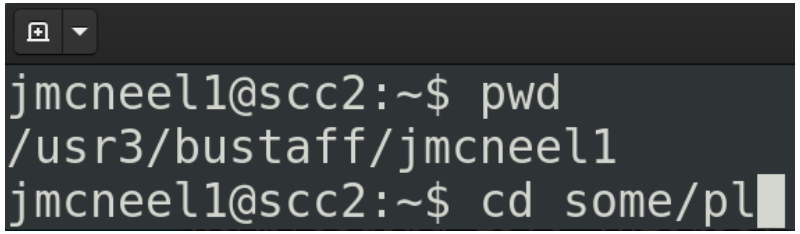
Tabbing Around:



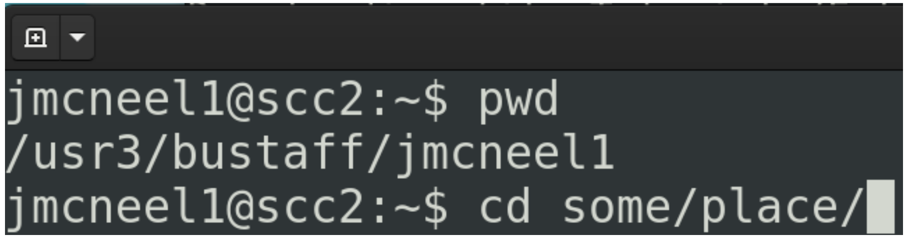
```
jmcneel1@scc2:~$ pwd
/usr3/bustaff/jmcneel1
jmcneel1@scc2:~$ cd so
```



```
jmcneel1@scc2:~$ pwd
/usr3/bustaff/jmcneel1
jmcneel1@scc2:~$ cd some/
```



```
jmcneel1@scc2:~$ pwd
/usr3/bustaff/jmcneel1
jmcneel1@scc2:~$ cd some/pl
```

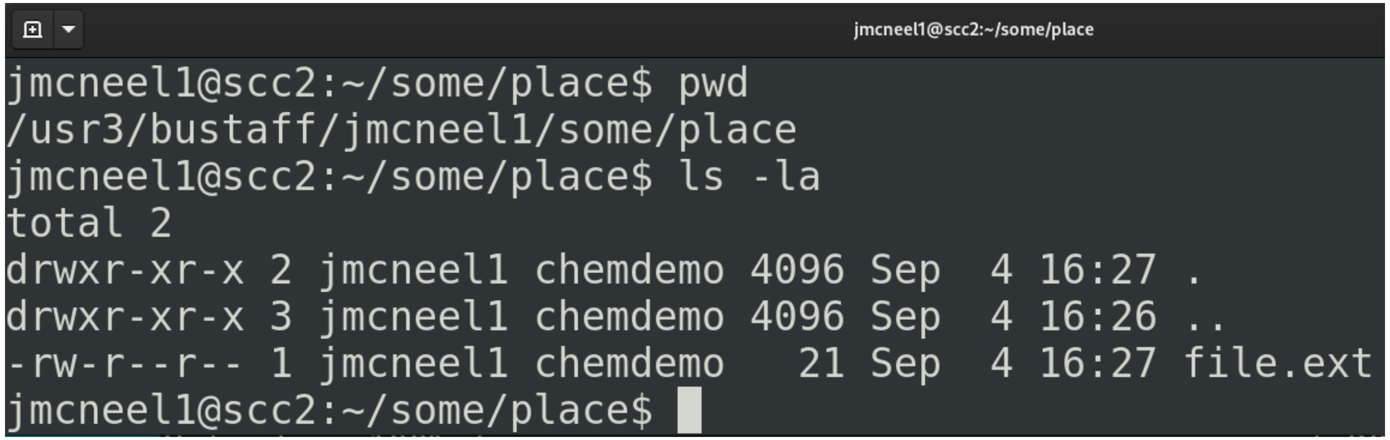


```
jmcneel1@scc2:~$ pwd
/usr3/bustaff/jmcneel1
jmcneel1@scc2:~$ cd some/place/
```

What’s in your pwd?

To see what files/folders are in your pwd:

- ls –la



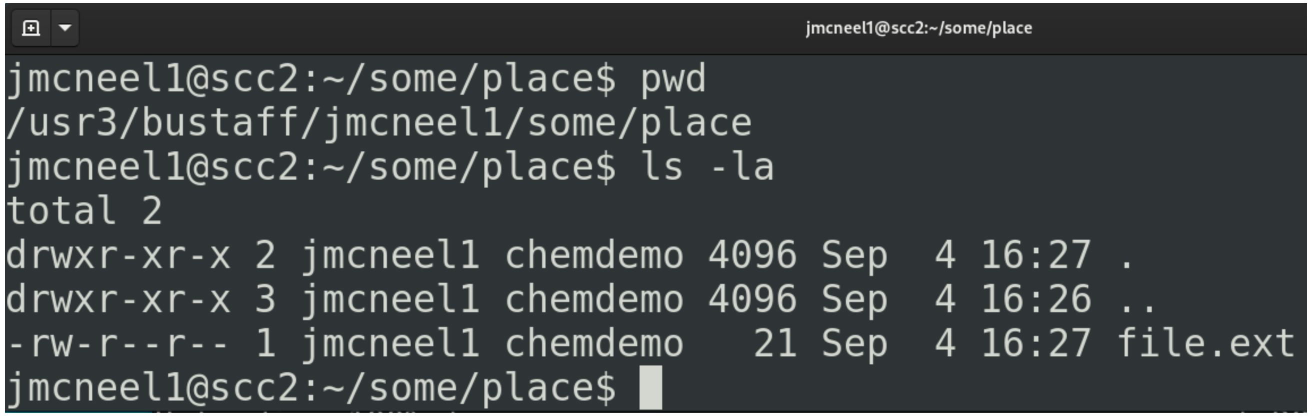
```
jmcneel1@scc2:~/some/place$ pwd
/usr3/bustaff/jmcneel1/some/place
jmcneel1@scc2:~/some/place$ ls -la
total 2
drwxr-xr-x 2 jmcneel1 chemdemo 4096 Sep  4 16:27 .
drwxr-xr-x 3 jmcneel1 chemdemo 4096 Sep  4 16:26 ..
-rw-r--r-- 1 jmcneel1 chemdemo   21 Sep  4 16:27 file.ext
jmcneel1@scc2:~/some/place$
```

Here we used a new command: ls

- ls: **L**i**S**t files/directories
- -la arguments tells command to also show hidden files and use a long listing format

I bequeath to thee Rhyme Five:

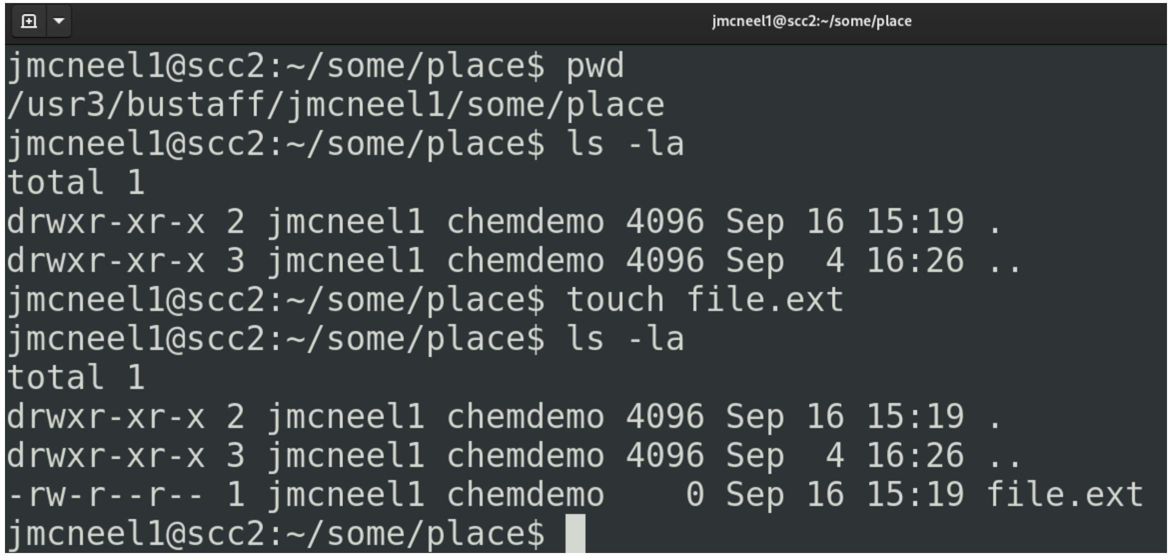
To see what files you can access, use the common command ls.



```
jmcneel1@scc2:~/some/place$ pwd
/usr3/bustaff/jmcneel1/some/place
jmcneel1@scc2:~/some/place$ ls -la
total 2
drwxr-xr-x 2 jmcneel1 chemdemo 4096 Sep  4 16:27 .
drwxr-xr-x 3 jmcneel1 chemdemo 4096 Sep  4 16:26 ..
-rw-r--r-- 1 jmcneel1 chemdemo   21 Sep  4 16:27 file.ext
jmcneel1@scc2:~/some/place$ 
```

At this point I’d for each of you to create your own directories at “~/some/place” using **mkdir –p** and then change your pwd to that directory using **cd** (if you haven’t already done so). Then check to see if it’s empty with **ls -la**:

If it’s empty (only ‘.’, and ‘..’ present), then create a file called file.ext using the command touch



```
jmcneel1@scc2:~/some/place$ pwd
/usr3/bustaff/jmcneel1/some/place
jmcneel1@scc2:~/some/place$ ls -la
total 1
drwxr-xr-x 2 jmcneel1 chemdemo 4096 Sep 16 15:19 .
drwxr-xr-x 3 jmcneel1 chemdemo 4096 Sep  4 16:26 ..
jmcneel1@scc2:~/some/place$ touch file.ext
jmcneel1@scc2:~/some/place$ ls -la
total 1
drwxr-xr-x 2 jmcneel1 chemdemo 4096 Sep 16 15:19 .
drwxr-xr-x 3 jmcneel1 chemdemo 4096 Sep  4 16:26 ..
-rw-r--r-- 1 jmcneel1 chemdemo    0 Sep 16 15:19 file.ext
jmcneel1@scc2:~/some/place$ 
```

Here we used a new command: touch

- touch: put your fingerprints on a file
- create an empty file that you own

Rhyme Six ensues:

With a single command, you can do so much –
create a file instantly, just by typing touch.

Placing content into a file

There are two easy options to place content into a file (among many other options):

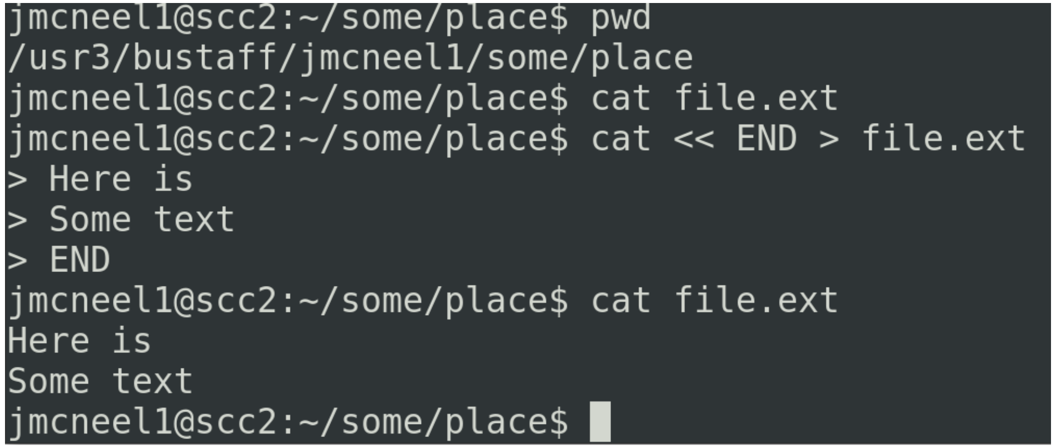
- The redirect operators '>' and '>>'
  - "Redirects" what would have been printed to the screen into a file

- Use the text editor 'nano'
  - A minimal text editor (think Notepad or TextEdit)

Redirecting into a file

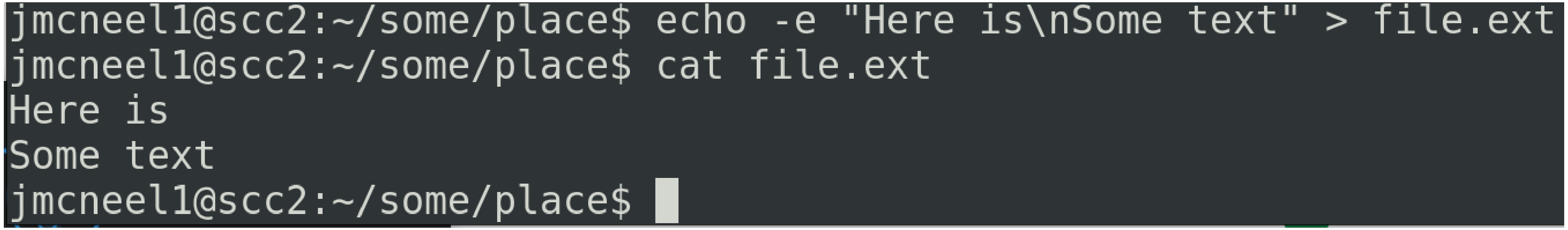
We can use commands we’ve already learned to redirect the command line results into a file:

- cat can easily be used to type text into a file



```
jmcneel1@scc2:~/some/place$ pwd
/usr3/bustaff/jmcneel1/some/place
jmcneel1@scc2:~/some/place$ cat file.ext
jmcneel1@scc2:~/some/place$ cat << END > file.ext
> Here is
> Some text
> END
jmcneel1@scc2:~/some/place$ cat file.ext
Here is
Some text
jmcneel1@scc2:~/some/place$ 
```

- And also a new command ‘echo’



```
jmcneel1@scc2:~/some/place$ echo -e "Here is\nSome text" > file.ext
jmcneel1@scc2:~/some/place$ cat file.ext
Here is
Some text
jmcneel1@scc2:~/some/place$ 
```

Here we used a new command: echo

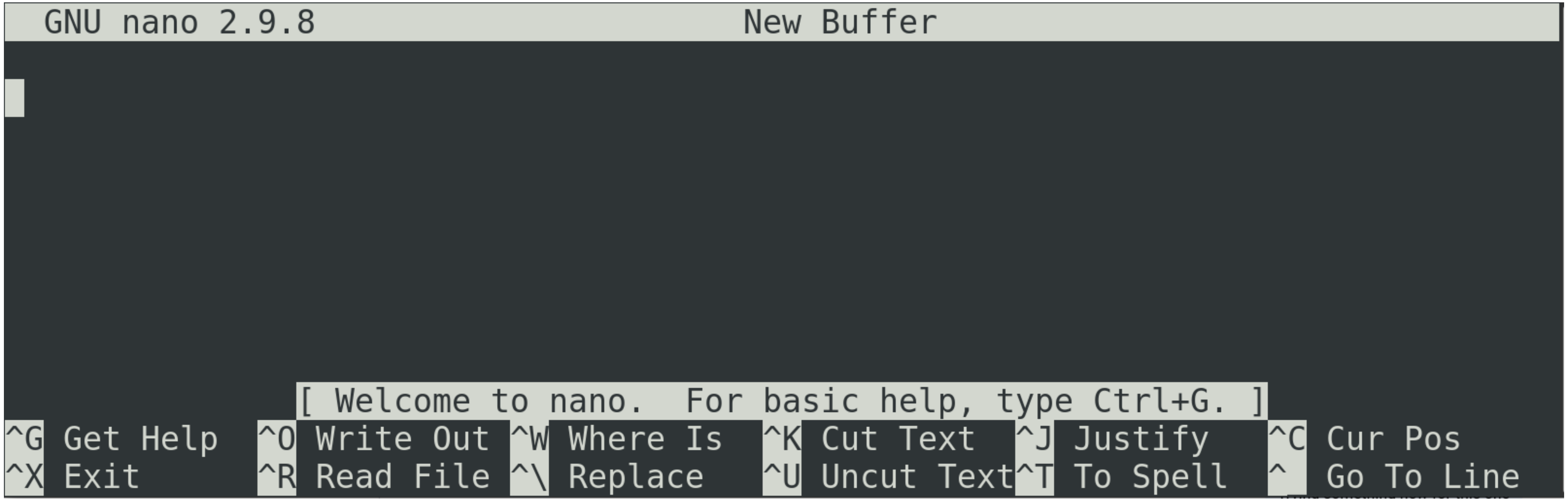
- echo: print a string or variable to the screen (echo it).

Rhyme Seven arrives:

If you want to allow variables and strings to show, just use the handy command echo.

Use the text editor ‘nano’

To open nano, simply type nano in the cmd line.
The main screen shows the keyboard shortcuts:



```
  GNU nano 2.9.8                        New Buffer




               [ Welcome to nano.  For basic help, type Ctrl+G. ]
^G Get Help   ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos
^X Exit       ^R Read File  ^\ Replace    ^U Uncut Text^T To Spell    ^  Go To Line
```

Copying/Moving/Renaming Files

Often times, instead of writing a file from scratch, we simply want to slightly modify a file already present on the filesystem:

- cp copies a file from one location to a new location

- mv moves a file from one location to a new location, or 'renames' it

Time for rhymes 8 and 9:

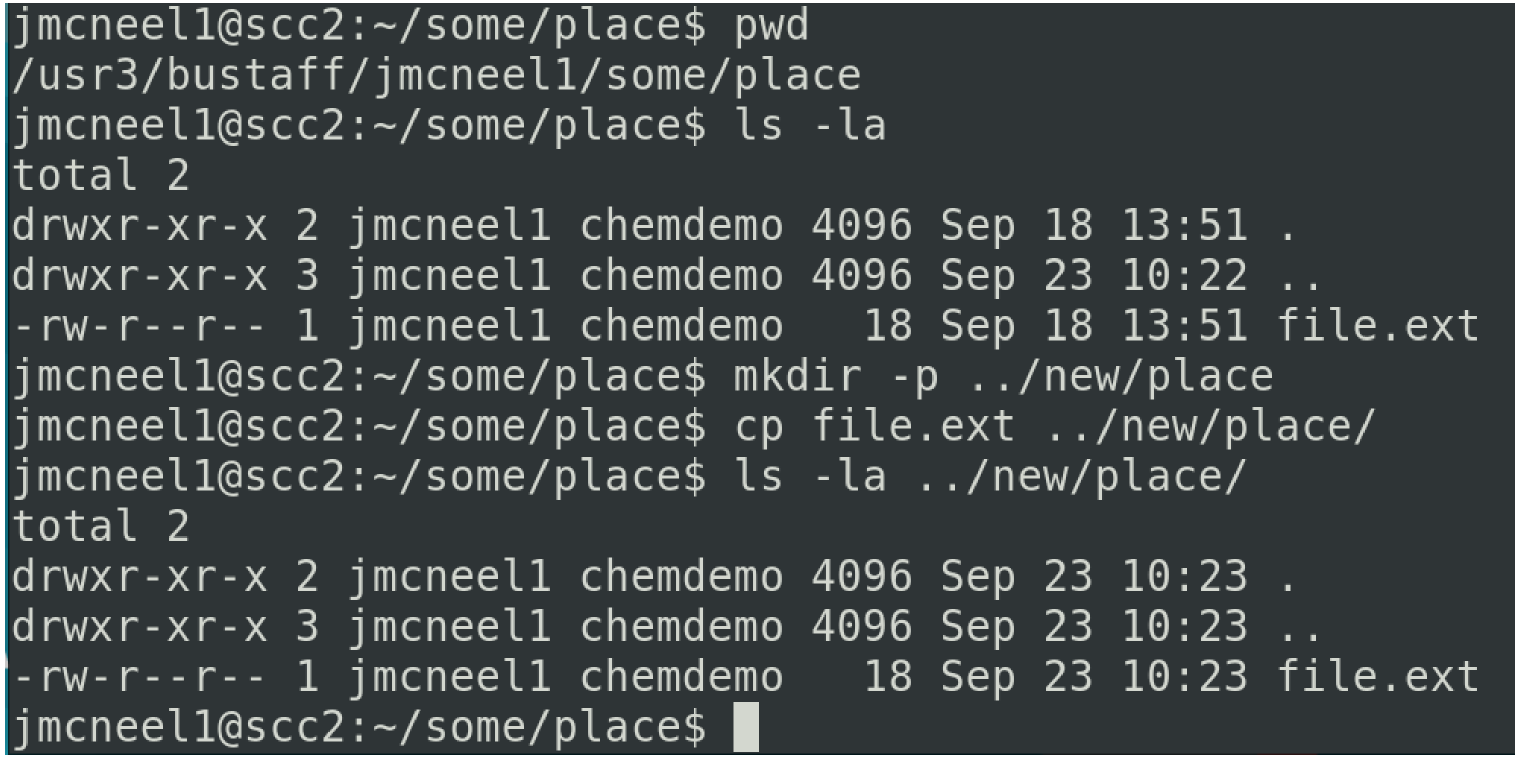
cp

To have a file or folder duplicated to a new folder on our PC we simply have to use the command cp.

mv

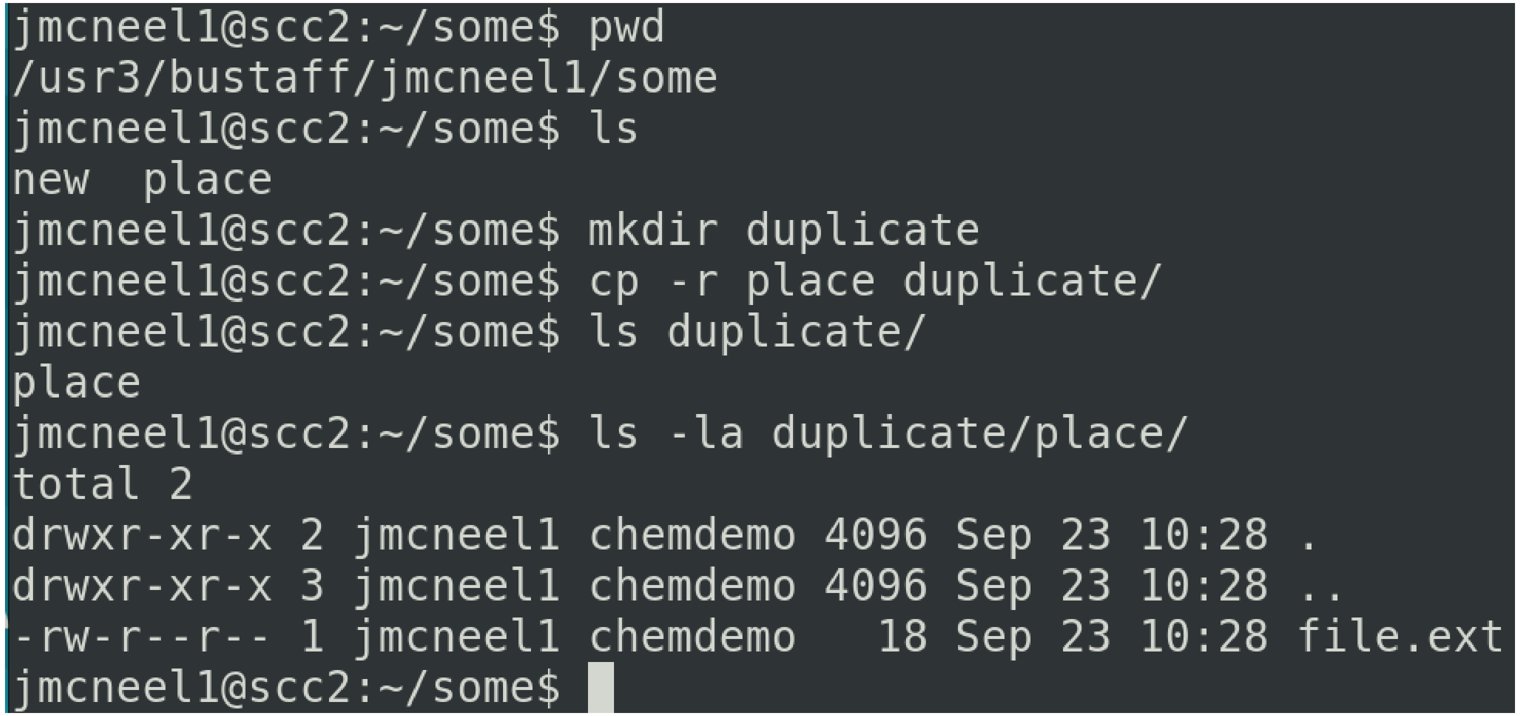
For a change of location of a file that we see, use the handy util mv.

Copying Files



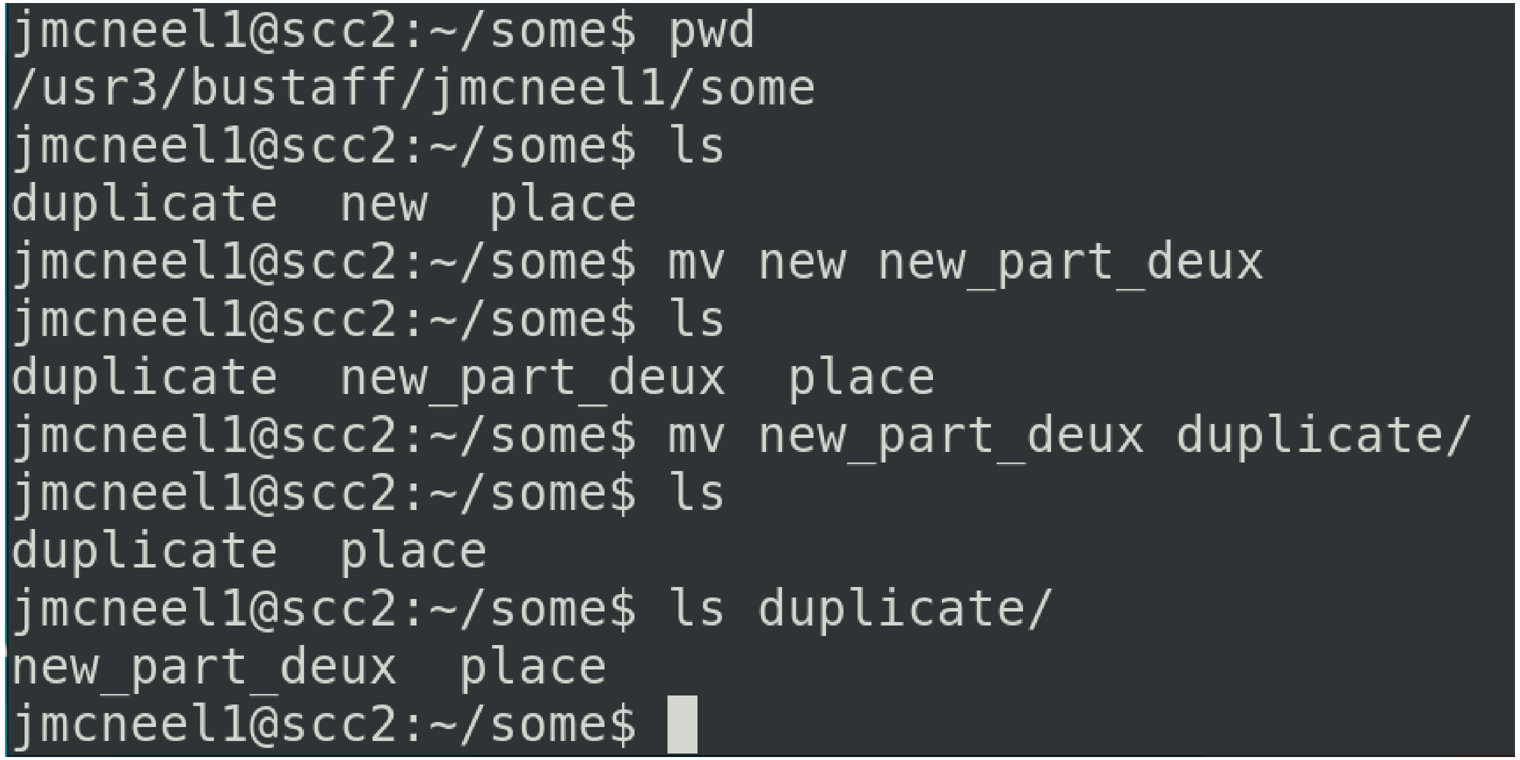
```
jmcneel1@scc2:~/some/place$ pwd
/usr3/bustaff/jmcneel1/some/place
jmcneel1@scc2:~/some/place$ ls -la
total 2
drwxr-xr-x 2 jmcneel1 chemdemo 4096 Sep 18 13:51 .
drwxr-xr-x 3 jmcneel1 chemdemo 4096 Sep 23 10:22 ..
-rw-r--r-- 1 jmcneel1 chemdemo   18 Sep 18 13:51 file.ext
jmcneel1@scc2:~/some/place$ mkdir -p ../new/place
jmcneel1@scc2:~/some/place$ cp file.ext ../new/place/
jmcneel1@scc2:~/some/place$ ls -la ../new/place/
total 2
drwxr-xr-x 2 jmcneel1 chemdemo 4096 Sep 23 10:23 .
drwxr-xr-x 3 jmcneel1 chemdemo 4096 Sep 23 10:23 ..
-rw-r--r-- 1 jmcneel1 chemdemo   18 Sep 23 10:23 file.ext
jmcneel1@scc2:~/some/place$ 
```

Copying Folders



```
jmcneel1@scc2:~/some$ pwd
/usr3/bustaff/jmcneel1/some
jmcneel1@scc2:~/some$ ls
new  place
jmcneel1@scc2:~/some$ mkdir duplicate
jmcneel1@scc2:~/some$ cp -r place duplicate/
jmcneel1@scc2:~/some$ ls duplicate/
place
jmcneel1@scc2:~/some$ ls -la duplicate/place/
total 2
drwxr-xr-x 2 jmcneel1 chemdemo 4096 Sep 23 10:28 .
drwxr-xr-x 3 jmcneel1 chemdemo 4096 Sep 23 10:28 ..
-rw-r--r-- 1 jmcneel1 chemdemo   18 Sep 23 10:28 file.ext
jmcneel1@scc2:~/some$
```

Moving Files/Folders

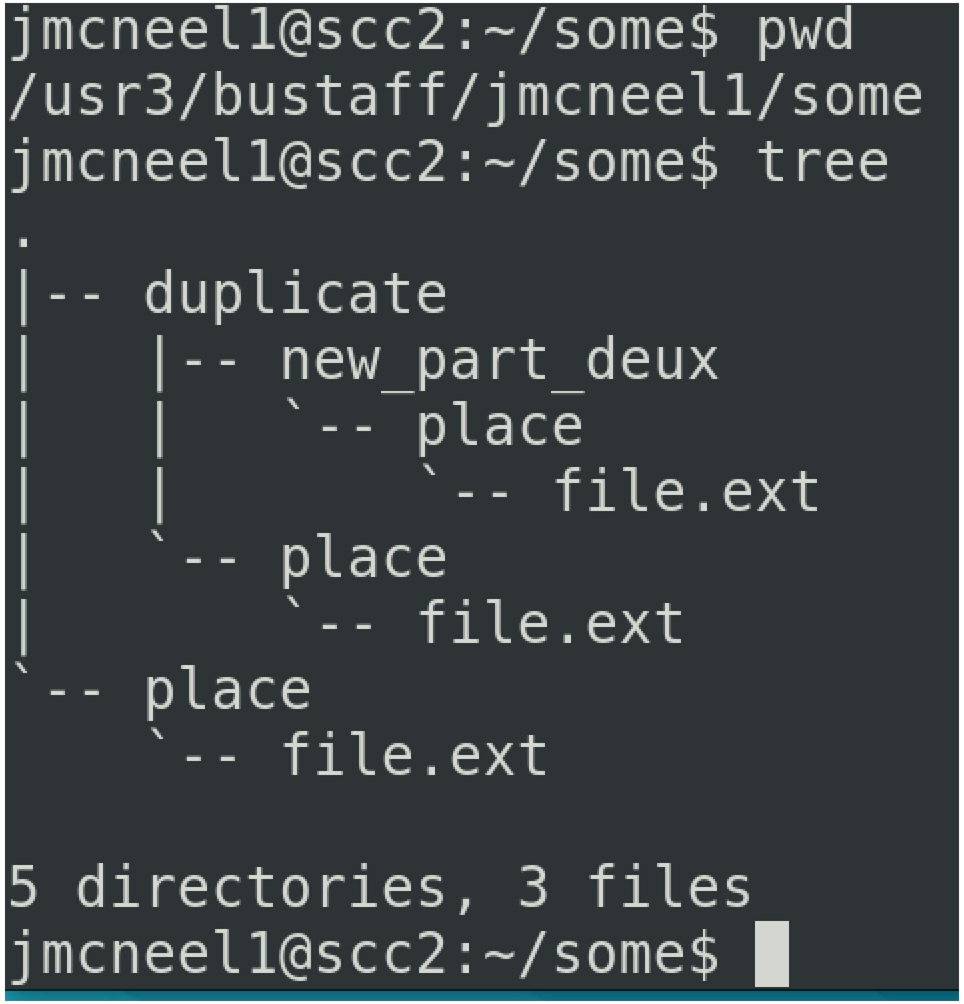


```
jmcneel1@scc2:~/some$ pwd
/usr3/bustaff/jmcneel1/some
jmcneel1@scc2:~/some$ ls
duplicate  new  place
jmcneel1@scc2:~/some$ mv new new_part_deux
jmcneel1@scc2:~/some$ ls
duplicate  new_part_deux  place
jmcneel1@scc2:~/some$ mv new_part_deux duplicate/
jmcneel1@scc2:~/some$ ls
duplicate  place
jmcneel1@scc2:~/some$ ls duplicate/
new_part_deux  place
jmcneel1@scc2:~/some$ 
```

Another New Command to Supplement ls

Sometimes we want to see what files/folders are in our pwd and also what is inside the folders:

- Tree shows the full content of your pwd



```
jmcneel1@scc2:~/some$ pwd
/usr3/bustaff/jmcneel1/some
jmcneel1@scc2:~/some$ tree
.
|-- duplicate
|   |-- new_part_deux
|   |   `-- place
|   |       `-- file.ext
|   `-- place
|       `-- file.ext
`-- place
    `-- file.ext

5 directories, 3 files
jmcneel1@scc2:~/some$
```

The boon of Rhyme 10:

tree:

For a view of all there is to see in your pwd, just enter the command tree.

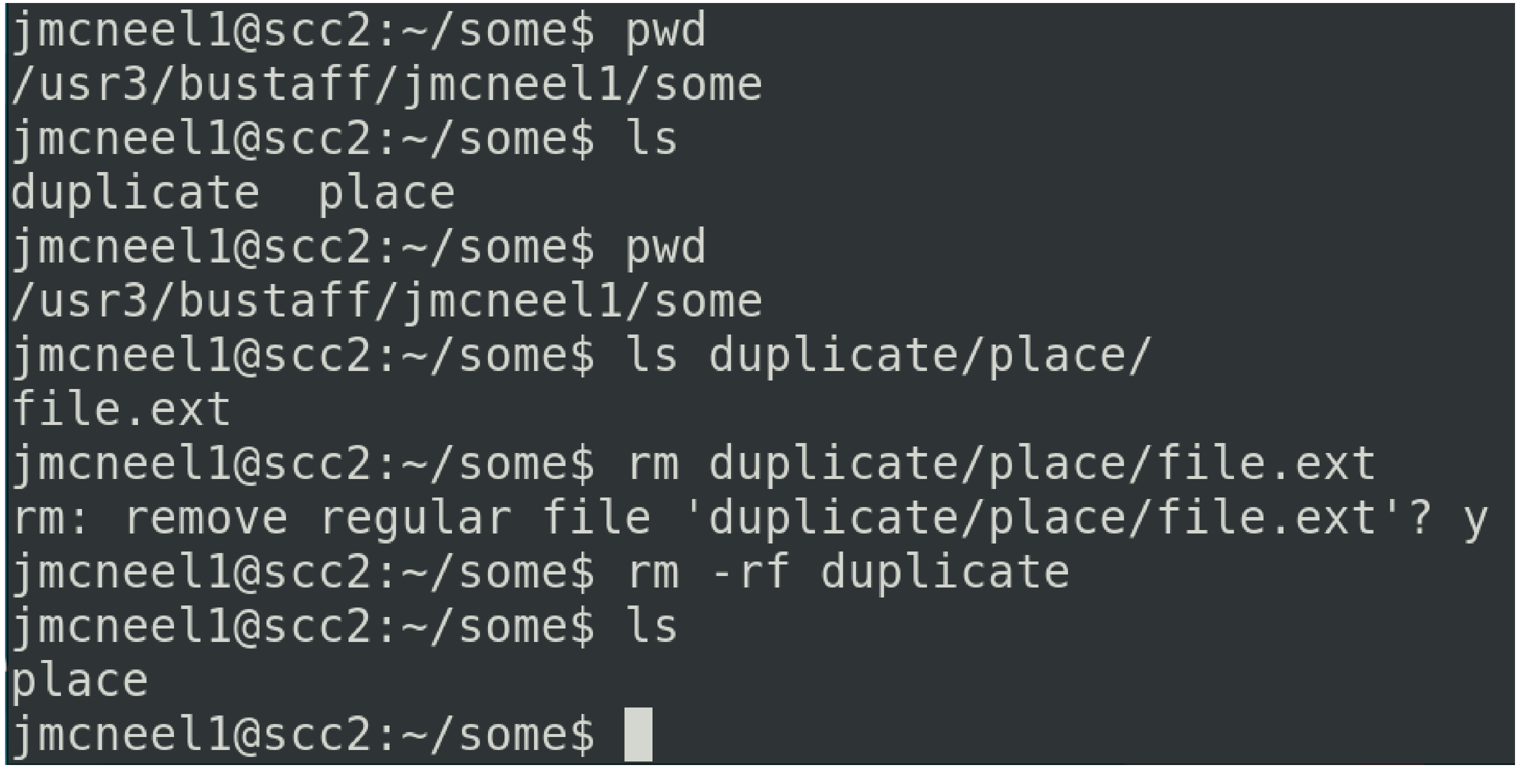
Deleting Files and Folders

rm: **R**e**M**ove

Deletion is accomplished like scissors to the stem with the mighty and dangerous command rm

- rm deletes file(s)
- rm *filename* prompts you before the deletion is performed
- rm –f *filename* DOESN’T prompt you fore the deletion is performed
- rm –rf *folder* DOESN’T prompt you and deletes a folder and all the content within

Deleting Files and Folders



```
jmcneel1@scc2:~/some$ pwd
/usr3/bustaff/jmcneel1/some
jmcneel1@scc2:~/some$ ls
duplicate  place
jmcneel1@scc2:~/some$ pwd
/usr3/bustaff/jmcneel1/some
jmcneel1@scc2:~/some$ ls duplicate/place/
file.ext
jmcneel1@scc2:~/some$ rm duplicate/place/file.ext
rm: remove regular file 'duplicate/place/file.ext'? y
jmcneel1@scc2:~/some$ rm -rf duplicate
jmcneel1@scc2:~/some$ ls
place
jmcneel1@scc2:~/some$ 
```

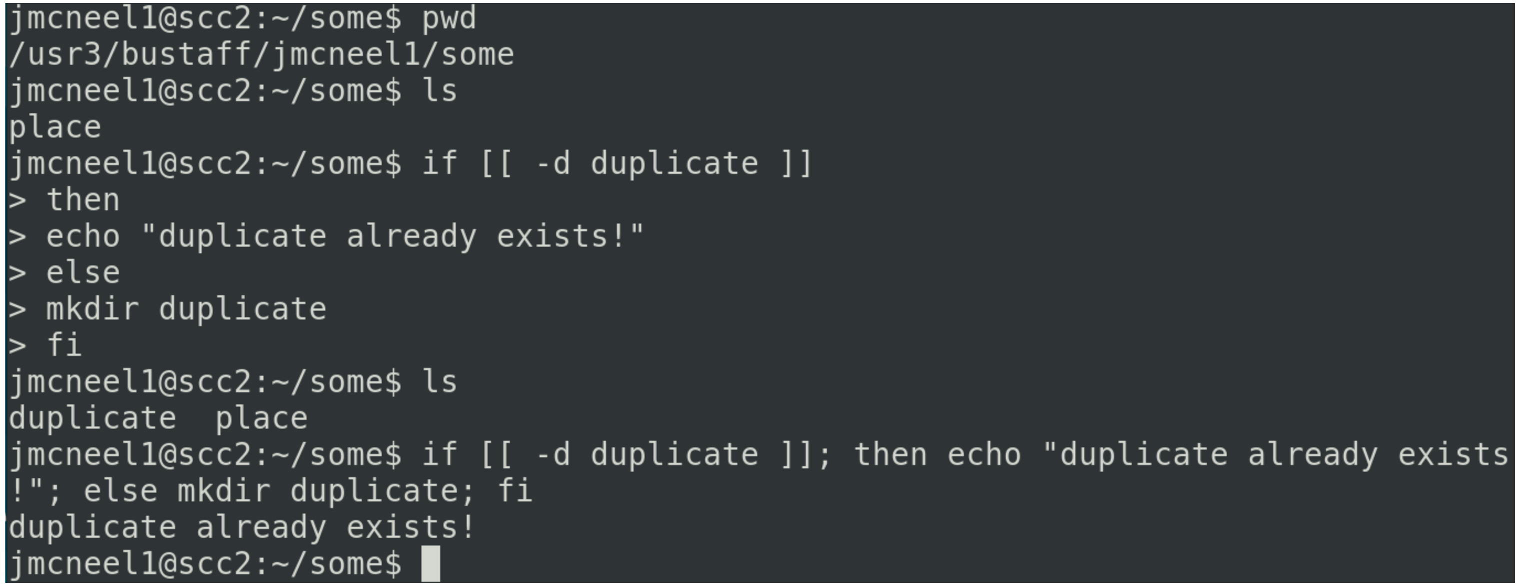
Control Structures: The if/then/else statement

```
if [[ condition is true ]]
then
do something
else
do something else
fi
```

conditions

- -d *name*: Directory exists
- -f *name*: File exists
- *s1* == *s2*: Strings equal
- *s1 != s2: String not equal*

Control Structures: The if/then/else statement



```
jmcneel1@scc2:~/some$ pwd
/usr3/bustaff/jmcneel1/some
jmcneel1@scc2:~/some$ ls
place
jmcneel1@scc2:~/some$ if [[ -d duplicate ]]
> then
> echo "duplicate already exists!"
> else
> mkdir duplicate
> fi
jmcneel1@scc2:~/some$ ls
duplicate  place
jmcneel1@scc2:~/some$ if [[ -d duplicate ]]; then echo "duplicate already exists
!"; else mkdir duplicate; fi
duplicate already exists!
jmcneel1@scc2:~/some$ 
```

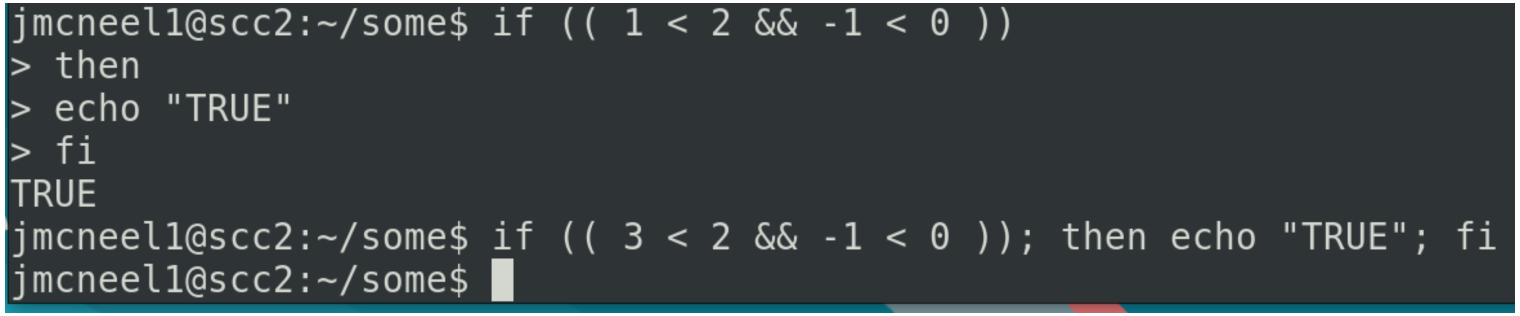
Control Structures: The if/then/else statement with arithmetic

```
if (( condition is true ))
then
do something
else
do something else
fi
```

conditions

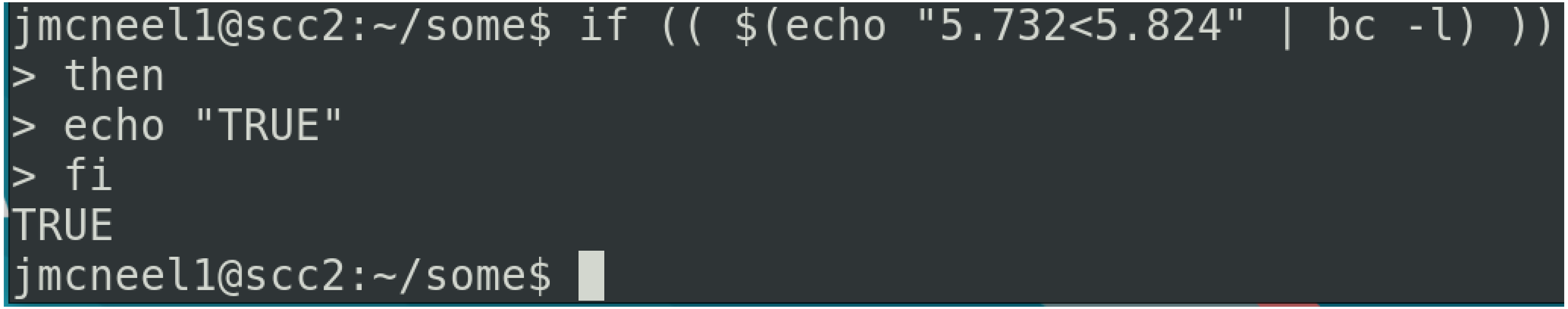
- ||: logical or
- &&: logical and
- ==, !=, <, <=, >, >=

Control Structures: The if/then/else statement with arithmetic



```
jmcneel1@scc2:~/some$ if (( 1 < 2 && -1 < 0 ))
> then
> echo "TRUE"
> fi
TRUE
jmcneel1@scc2:~/some$ if (( 3 < 2 && -1 < 0 )); then echo "TRUE"; fi
jmcneel1@scc2:~/some$ 
```

Control Structures: How about floating point numbers?



```
jmcneel1@scc2:~/some$ if (( $(echo "5.732<5.824" | bc -l) ))
> then
> echo "TRUE"
> fi
TRUE
jmcneel1@scc2:~/some$ 
```

Here we introduced 2 new things:

1. bc: **B**asic **C**alculator
2. $(*command*): Command Substitution

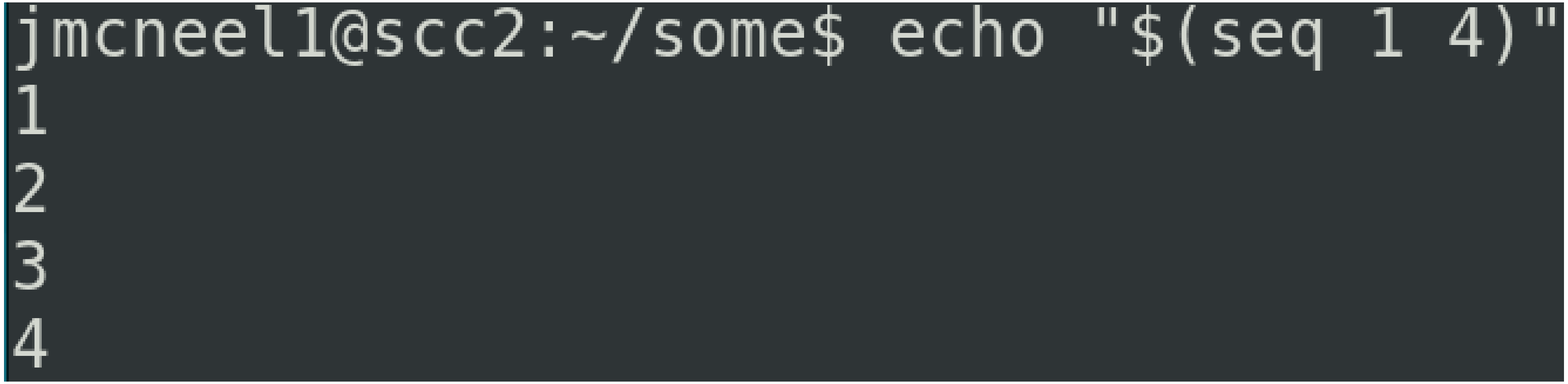
Rhyme 12 is in session:

bc:

To crunch some numbers with glee, use the command bc.

Command Substitution

- Allows the output of a command to replace the command itself.



```
jmcneel1@scc2:~/some$ echo "$(seq 1 4)"
1
2
3
4
```

- An alternative syntax is to use `*command*`
- Use $(()) for arithmetic

A New Command!

- seq: **SEQ**uence
- Returns integers between a start and end value increments of 1.

Rhyme Time (Number 13)!

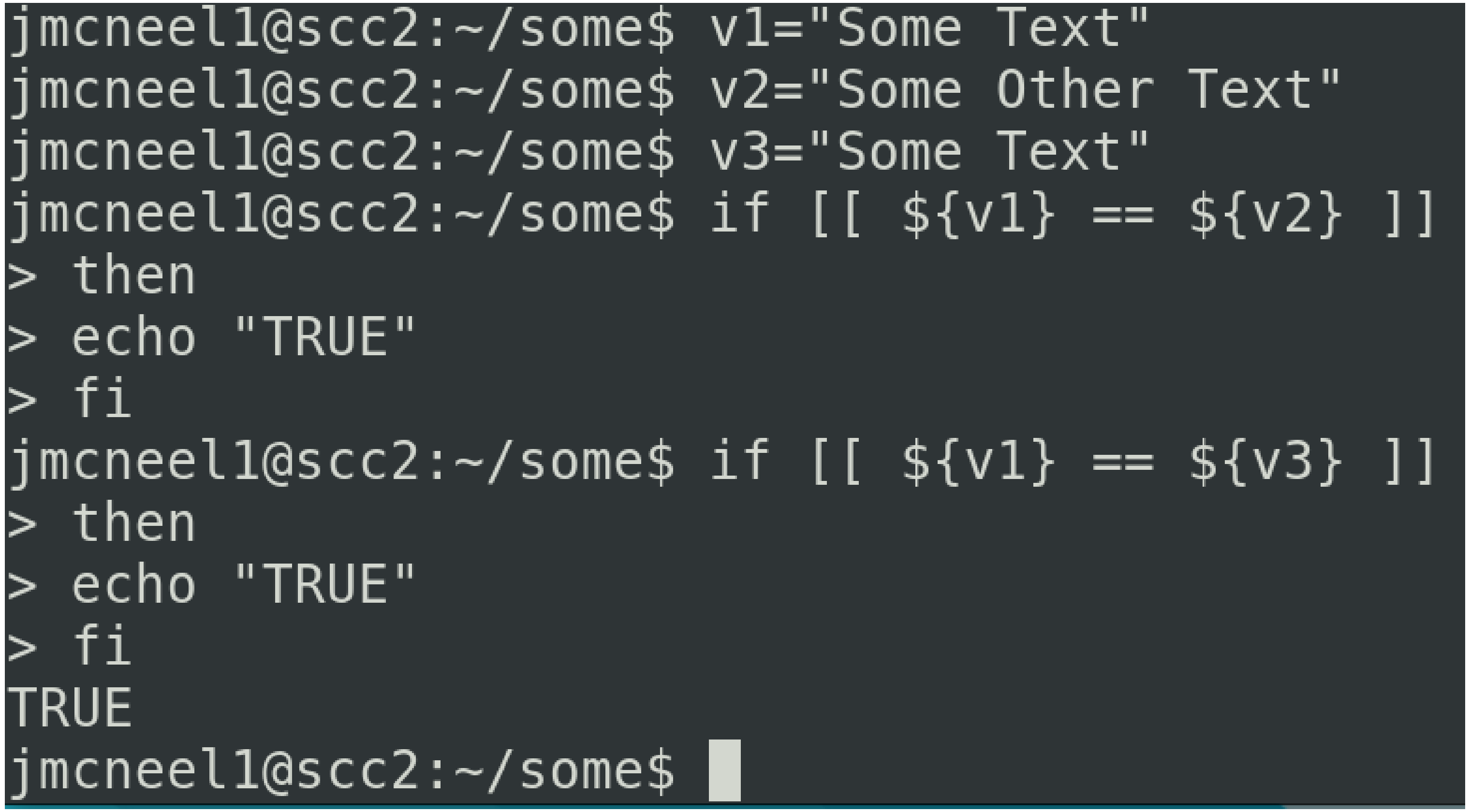
seq:

To incrementally move from the start to desuetude, go ahead and type in seq.

Variables

- In ALL programming, a variable is a name that refers to a location on the computer memory that contains information.
- We use variables to store information
- We use variables to alter information
- We use variables to probe multiple pieces of information

Variables: In Bash, all variables are strings

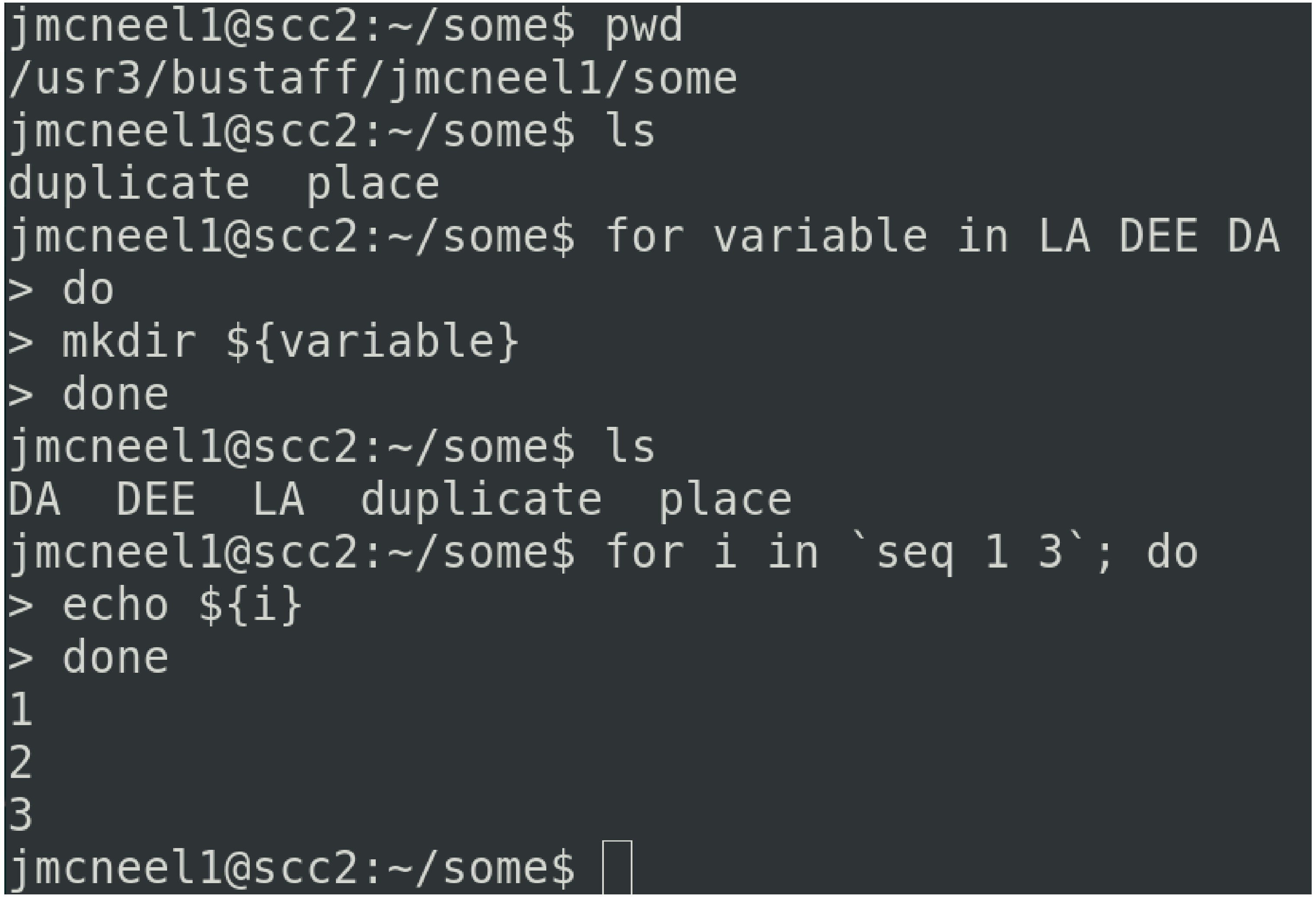


```
jmcneel1@scc2:~/some$ v1="Some Text"
jmcneel1@scc2:~/some$ v2="Some Other Text"
jmcneel1@scc2:~/some$ v3="Some Text"
jmcneel1@scc2:~/some$ if [[ ${v1} == ${v2} ]]
> then
> echo "TRUE"
> fi
jmcneel1@scc2:~/some$ if [[ ${v1} == ${v3} ]]
> then
> echo "TRUE"
> fi
TRUE
jmcneel1@scc2:~/some$ 
```

Looping: the for loop

- Now for one of the most useful commands in bash (or nearly any computer language). The for loop.
- For loops iterate through every element of a list and performs an identical operation.
- If a list has 100 items, instead of typing the operation command 100 times, enter it ONCE inside a for loop.

For Loops

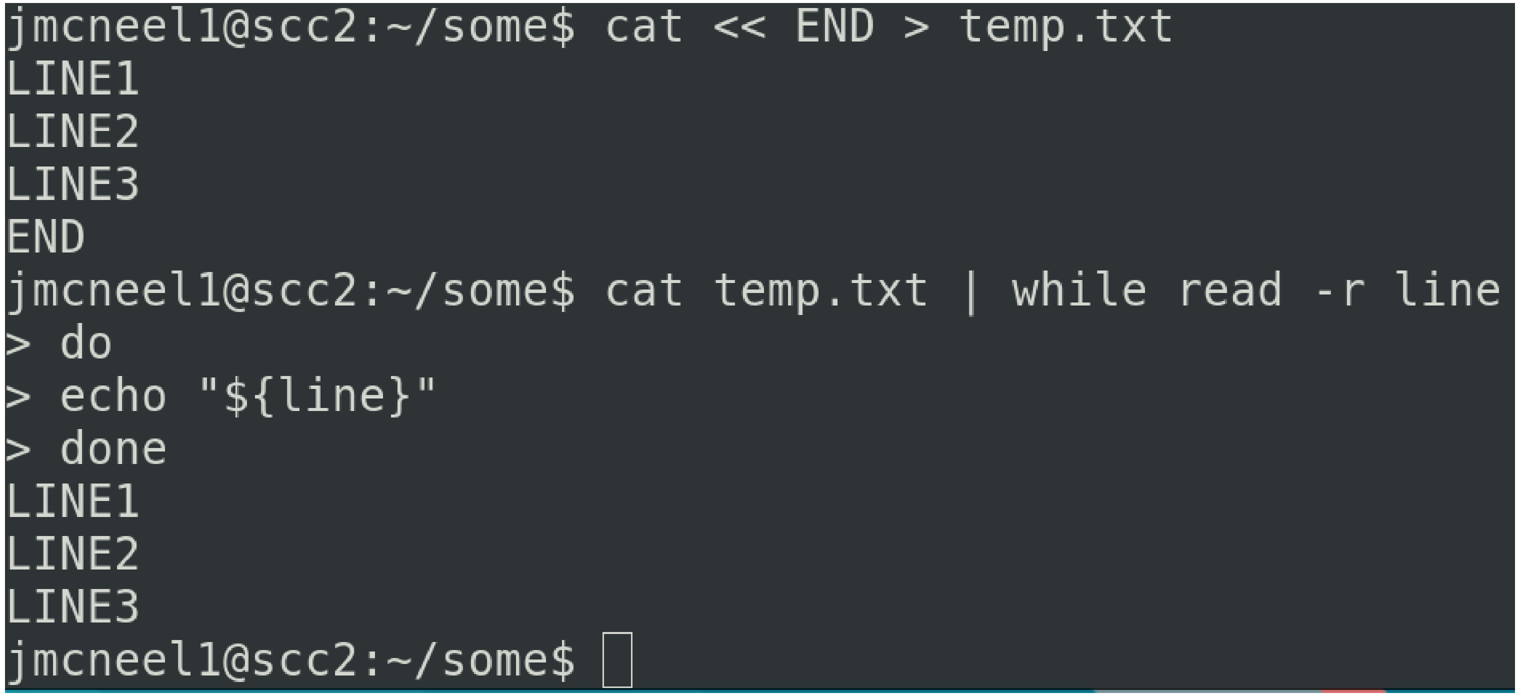


```
jmcneel1@scc2:~/some$ pwd
/usr3/bustaff/jmcneel1/some
jmcneel1@scc2:~/some$ ls
duplicate  place
jmcneel1@scc2:~/some$ for variable in LA DEE DA
> do
> mkdir ${variable}
> done
jmcneel1@scc2:~/some$ ls
DA  DEE  LA  duplicate  place
jmcneel1@scc2:~/some$ for i in `seq 1 3`; do
> echo ${i}
> done
1
2
3
jmcneel1@scc2:~/some$ 
```

Looping: the while loop

- While loops are preferred to for loops when you don’t know in advance how many times the loop will run…
- Very useful for reading files.
- Can also be used with conditional expressions.

While Loop: Reading a File



```
jmcneel1@scc2:~/some$ cat << END > temp.txt
LINE1
LINE2
LINE3
END
jmcneel1@scc2:~/some$ cat temp.txt | while read -r line
> do
> echo "${line}"
> done
LINE1
LINE2
LINE3
jmcneel1@scc2:~/some$ 
```

Here we used a new operator

- ‘|’: a “pipe”

- Pipes the results of the lhs expression into the rhs

- Very useful with some commands like ‘grep’ and ‘awk’

A New Command!

- grep: **G**lobal **R**egular **E**xpression search and **P**rint
- Searches an argument file or string for a search term.

Rhymin' Simon Number 14:

grep:

To see if some text is in my pipe or file no need to fret,
just use the awesome command grep.

A New Command!

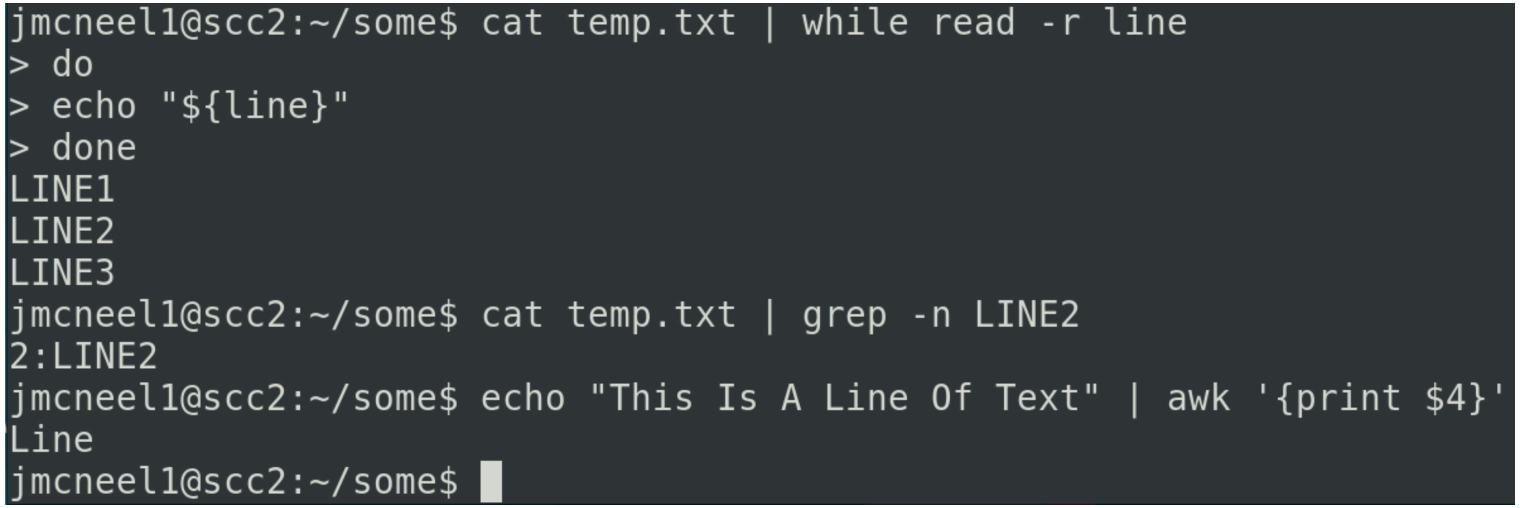
- awk: **A**ho, **W**einberger, and **K**ernighan
- Basically another language.
- We’ll only use it here to do the most simple of functions:

Rhyme 15 is a doozy:

awk:

To perform advanced splitting of strings you'll be shocked, this magic command that those in the know call awk.

Piping with grep and awk



```
jmcneel1@scc2:~/some$ cat temp.txt | while read -r line
> do
> echo "${line}"
> done
LINE1
LINE2
LINE3
jmcneel1@scc2:~/some$ cat temp.txt | grep -n LINE2
2:LINE2
jmcneel1@scc2:~/some$ echo "This Is A Line Of Text" | awk '{print $4}'
Line
jmcneel1@scc2:~/some$ 
```

The Final Command: sed

- sed: **S**tream **E**ditor
- This is a command to alter text in a stream.

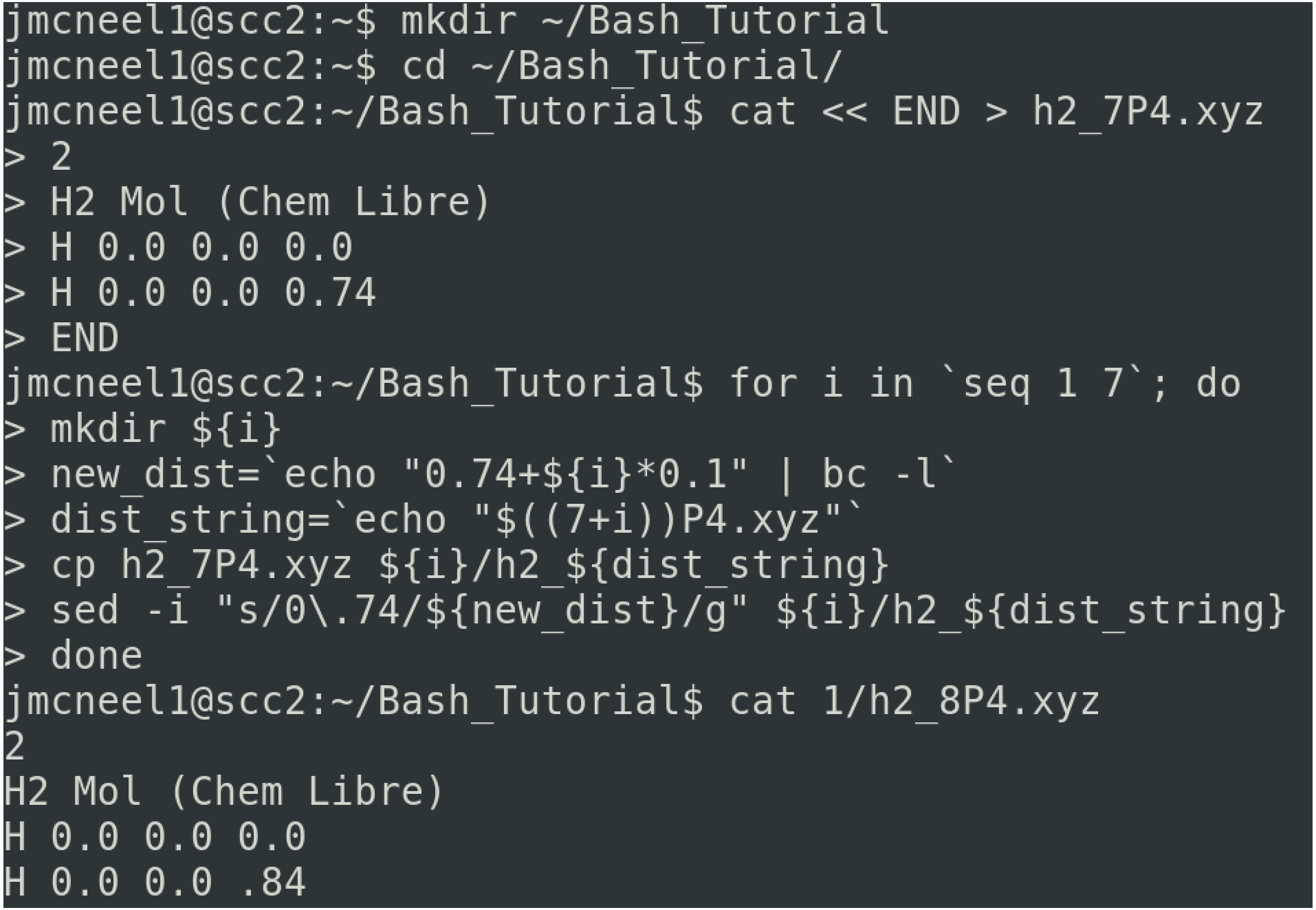
Rhyme 16 at last:

sed:

To edit text in a stream, before we put our command list to bed, Bell Labs gave you sed.

Now Let’s Do Something ‘Real’

- Create a Folder ~/Bash_Tutorial
- Change your PWD to this folder
- Create an XYZ file containing the XYZ coordinates for a hydrogen molecule and place it into a file.
- Now create a looping structure that changes the H-H bond length to values of 0.84-1.54 Å in steps of 0.1 Å. Place each “new” XYZ file in its own folder



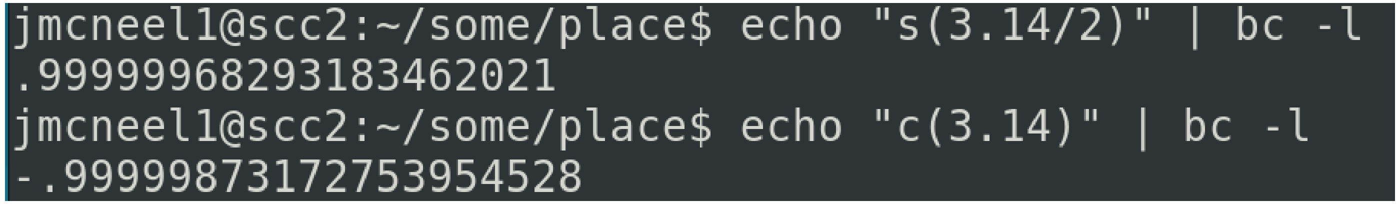
```
jmcneel1@scc2:~$ mkdir ~/Bash_Tutorial
jmcneel1@scc2:~$ cd ~/Bash_Tutorial/
jmcneel1@scc2:~/Bash_Tutorial$ cat << END > h2_7P4.xyz
> 2
> H2 Mol (Chem Libre)
> H 0.0 0.0 0.0
> H 0.0 0.0 0.74
> END
jmcneel1@scc2:~/Bash_Tutorial$ for i in `seq 1 7`; do
> mkdir ${i}
> new_dist=`echo "0.74+${i}*0.1" | bc -l`
> dist_string=`echo "$((7+i))P4.xyz"`
> cp h2_7P4.xyz ${i}/h2_${dist_string}
> sed -i "s/0\.74/${new_dist}/g" ${i}/h2_${dist_string}
> done
jmcneel1@scc2:~/Bash_Tutorial$ cat 1/h2_8P4.xyz
2
H2 Mol (Chem Libre)
H 0.0 0.0 0.0
H 0.0 0.0 .84
```

For You

- Do a similar operation, but change the H-O-H angle in water while keeping the bond lengths constant.

$$x_2 = x_1 cos(\Delta\theta) + y_1 sin(\Delta\theta)$$
$$y_2 = -x_1 sin(\Delta\theta) + y_1 cos(\Delta\theta)$$

```
3
Water molecule
O      0.00000     0.00000     0.11779
H      0.00000     0.75545    -0.47116
H      0.00000    -0.75545    -0.47116
```



```
jmcneel1@scc2:~/some/place$ echo "s(3.14/2)" | bc -l
.99999968293183462021
jmcneel1@scc2:~/some/place$ echo "c(3.14)" | bc -l
-.99999873172753954528
```

```
jmcneel1@scc2:~/some/place$ echo "sqrt((0.47116+0.11779)^2+0.75545^2)" | bc -l
.95789707432479404143
```