

FOAR705 - Learning Journal Week 6 - Software Carpentry Exercises

Jan Jugueta - 44828020

Week 6: 30/8/19 - 6/9/19

30/8/19 - 2:17pm

Going through some Software Carpentry exercises in the FOAR705 class. Some useful commands:

`Ctrl + C`

which is cancel.

`Ctrl + D`

which is disconnect.

30/8/19 - 2:37pm

Still practicing some Shell things. Practicing how to navigate again.

```
Last login: Fri Aug 30 14:06:43 on ttys000
[Janakins-MacBook-Pro:~ janakin$ %1
-bash: fg: %1: no such job
[Janakins-MacBook-Pro:~ janakin$ nano
[Janakins-MacBook-Pro:~ janakin$ cd
[Janakins-MacBook-Pro:~ janakin$ cd D
/Desktop/ Documents/ Downloads/
[Janakins-MacBook-Pro:~ janakin$ cd D
/Desktop/ Documents/ Downloads/
[Janakins-MacBook-Pro:~ janakin$ cd Downloads/
[Janakins-MacBook-Pro:~ janakin$ cd Desktop/
[Janakins-MacBook-Pro:Desktop janakin$ ls
190311 NEC 015 - Sydney FC 015 - Original.mp4
190311 NEC 017 - Sydney FC 017 - Original.mp4
IMG_3666.jpg
IMG_3695.jpg
MVI_0181.MP4
Nikki
Nikki USB
Player Interviews
Player Interviews.zip
Screen Shot 2019-08-23 at 11.20.21 am.png
Screen Shot 2019-08-29 at 1.58.58 pm.png
Screen Shot 2019-08-29 at 11.04.32 pm.png
data-shell
dogs
franky.jpg
[Janakins-MacBook-Pro:Desktop janakin$ cd data-shell/
[Janakins-MacBook-Pro:data-shell janakin$ cd
[Janakins-MacBook-Pro:~ janakin$ cd Desktop/data-shell/
[Janakins-MacBook-Pro:data-shell janakin$ ls
creatures          notes.txt          thesis_backup
data               pizza.cfg          writing
molecules          solar.pdf
north-pacific-gyre thesis
[Janakins-MacBook-Pro:data-shell janakin$ ls molecules/
cubane.pdb        methane.pdb      pentane.pdb
ethane.pdb        octane.pdb       propane.pdb
[Janakins-MacBook-Pro:data-shell janakin$ cd molecules/
Janakins-MacBook-Pro:molecules janakin$ ]
```

30/8/19 - 2:43pm

The command

```
history
```

shows all the commands you have used previously.

The command

```
>
```

redirects.

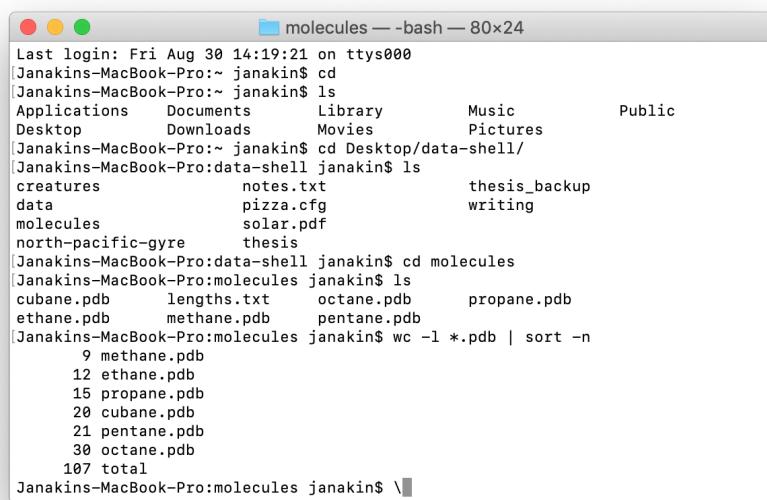
The command

```
cat
```

opens files.

30/8/19 - 2:58pm

My first pipe attempt.



A screenshot of a macOS terminal window titled "molecules -- bash -- 80x24". The window shows a command-line session:

```
Last login: Fri Aug 30 14:19:21 on ttys000
[Janakins-MacBook-Pro:~ janakin$ cd
[Janakins-MacBook-Pro:~ janakin$ ls
Applications Documents Library Music Public
Desktop Downloads Movies Pictures
[Janakins-MacBook-Pro:~ janakin$ cd Desktop/data-shell/
[Janakins-MacBook-Pro:data-shell janakin$ ls
creatures notes.txt thesis_backup
data pizza.cfg writing
molecules solar.pdf
north-pacific-gyre thesis
[Janakins-MacBook-Pro:data-shell janakin$ cd molecules
[Janakins-MacBook-Pro:molecules janakin$ ls
cubane.pdb lengths.txt octane.pdb propane.pdb
ethane.pdb methane.pdb pentane.pdb
[Janakins-MacBook-Pro:molecules janakin$ wc -l *.pdb | sort -n
      9 methane.pdb
     12 ethane.pdb
     15 propane.pdb
     20 cubane.pdb
     21 pentane.pdb
    30 octane.pdb
  107 total
Janakins-MacBook-Pro:molecules janakin$ \|
```

4/9/19 - 12:38pm

Now time to do the SW Carpentry exercises properly. Continuing with Episode 4 **Pipes and Filters**

Have followed the instructions so far and no errors.

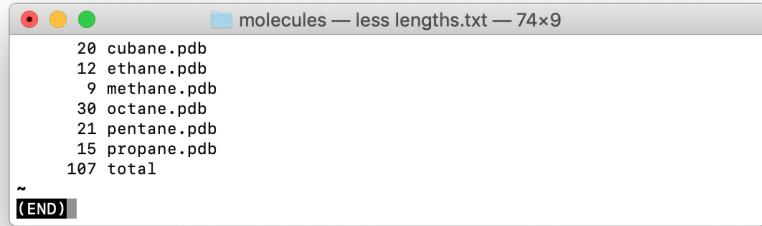
```
molecules -- bash -- 77x48
Last login: Wed Sep  4 12:39:57 on ttys000
[Janakins-MacBook-Pro:~ janakin$ ls
Applications      Documents      Library      Music      Public
Desktop          Downloads      Movies       Pictures
[Janakins-MacBook-Pro:~ janakin$ cd Desktop/data-shell/
[Janakins-MacBook-Pro:data-shell janakin$ ls
creatures          notes.txt      thesis_backup
data              pizza.cfg      writing
molecules          solar.pdf
north-pacific-gyre thesis
[Janakins-MacBook-Pro:data-shell janakin$ ls molecules
cubane.pdb        lengths.txt    octane.pdb    propane.pdb
ethane.pdb        methane.pdb   pentane.pdb
[Janakins-MacBook-Pro:data-shell janakin$ cd molecules
[Janakins-MacBook-Pro:molecules janakin$ wc *.pdb
 20      156     1158 cubane.pdb
 12       84      622 ethane.pdb
 9        57      422 methane.pdb
 30      246     1828 octane.pdb
 21      165     1226 pentane.pdb
 15      111      825 propane.pdb
 107     819     6081 total
[Janakins-MacBook-Pro:molecules janakin$ wc -l *.pdb
 20 cubane.pdb
 12 ethane.pdb
  9 methane.pdb
 30 octane.pdb
 21 pentane.pdb
 15 propane.pdb
 107 total
[Janakins-MacBook-Pro:molecules janakin$ wc -l
^C
[Janakins-MacBook-Pro:molecules janakin$ wc -l *.pdb > lengths.txt
[Janakins-MacBook-Pro:molecules janakin$ ls lengths.txt
lengths.txt
[Janakins-MacBook-Pro:molecules janakin$ cat lengths.txt
 20 cubane.pdb
 12 ethane.pdb
  9 methane.pdb
 30 octane.pdb
 21 pentane.pdb
 15 propane.pdb
 107 total
Janakins-MacBook-Pro:molecules janakin$
```

4/9/19 - 12:46pm

The

less

command in action.



```
20 cubane.pdb
12 ethane.pdb
 9 methane.pdb
30 octane.pdb
21 pentane.pdb
15 propane.pdb
107 total
~ (END)
```

4/9/19 - 12:49pm

The answer to the exercise *What does sort -n Do?* is that

```
sort -n
```

sorts purely numerically, not alphanumerically.

4/9/19 - 12:54pm

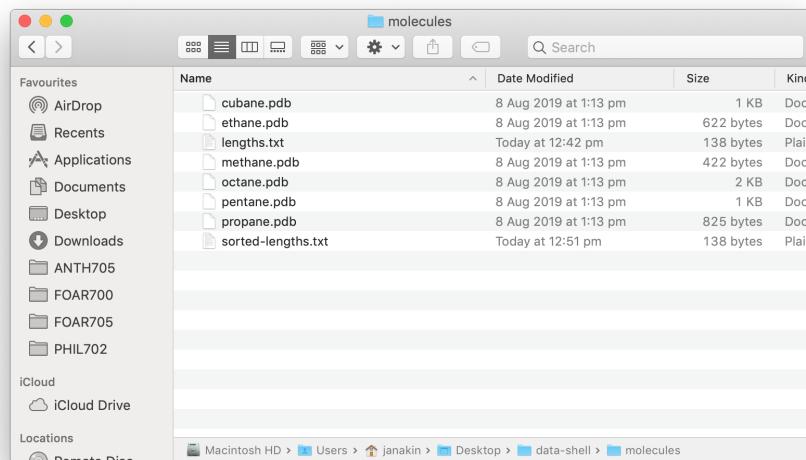
Continuing on with the Episode. The following screenshots show the

```
sort -n lengths.txt > sorted-lengths.txt
head -n 1 sorted-lengths.txt
```

commands.

```

|^CJanakins-MacBook-Pro:molecules janakin$ pwd
|/Users/janakin/Desktop/data-shell/molecules
|Janakins-MacBook-Pro:molecules janakin$ sort molecules
|sort: No such file or directory
|Janakins-MacBook-Pro:molecules janakin$ sort molecules
|Janakins-MacBook-Pro:molecules janakin$ pwd
|/Users/janakin/Desktop/data-shell/molecules
|Janakins-MacBook-Pro:molecules janakin$ sort -n lengths.txt
|      9 methane.pdb
|     12 ethane.pdb
|     15 propane.pdb
|     20 cubane.pdb
|     21 pentane.pdb
|    30 octane.pdb
|   107 total
|Janakins-MacBook-Pro:molecules janakin$ sort -n lengths.txt > sorted-lengths.txt
|Janakins-MacBook-Pro:molecules janakin$ head -n 1 sorted-lengths.txt
|      9 methane.pdb
Janakins-MacBook-Pro:molecules janakin$
```



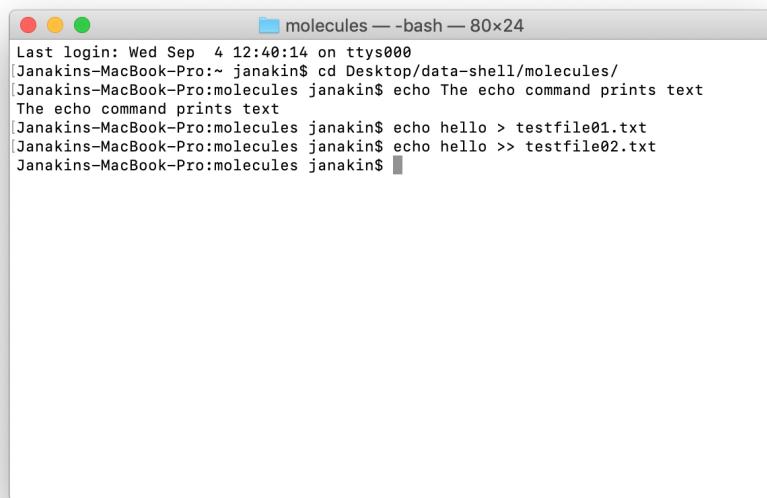
4/9/19 - 12:58pm

What does

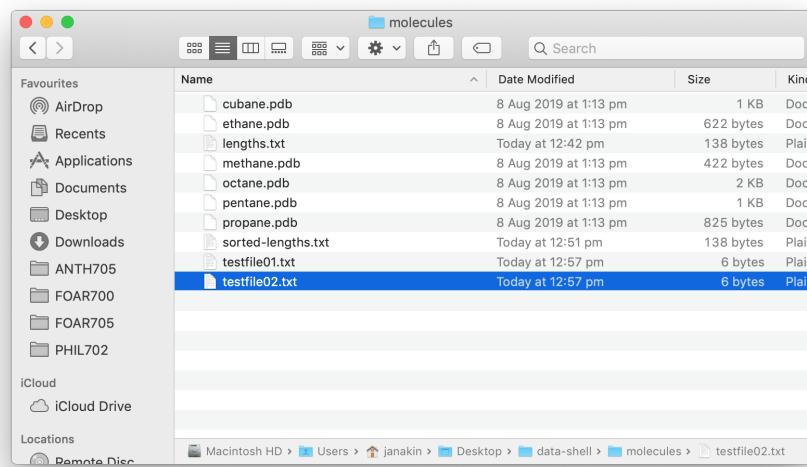
>>

mean?

Well, I've done exactly as the lesson says. But I still don't get it.



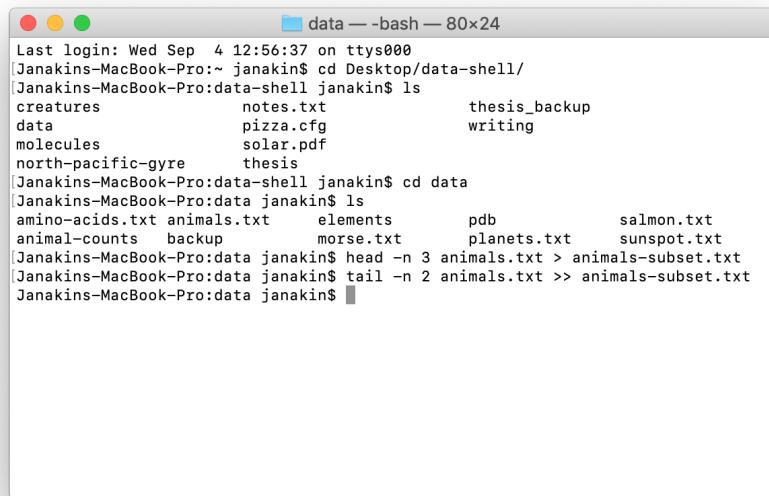
```
molecules -- bash -- 80x24
Last login: Wed Sep  4 12:40:14 on ttys000
[Janakins-MacBook-Pro:~ janakin$ cd Desktop/data-shell/molecules/
[Janakins-MacBook-Pro:molecules janakin$ echo The echo command prints text
The echo command prints text
[Janakins-MacBook-Pro:molecules janakin$ echo hello > testfile01.txt
[Janakins-MacBook-Pro:molecules janakin$ echo hello >> testfile02.txt
[Janakins-MacBook-Pro:molecules janakin$
```



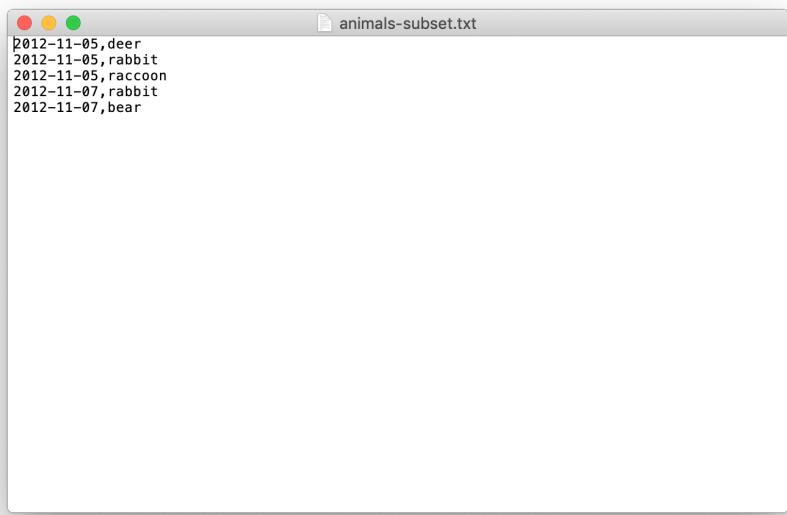
4/9/19 - 12:58pm

The *Appending Data* exercise.

Answer 3. It contains the first three lines and last 2 lines.



```
Last login: Wed Sep  4 12:56:37 on ttys000
[Janakins-MacBook-Pro:~ janakin$ cd Desktop/data-shell/
[Janakins-MacBook-Pro:data-shell janakin$ ls
creatures          notes.txt          thesis_backup
data               pizza.cfg          writing
molecules          solar.pdf
north-pacific-gyre thesis
[Janakins-MacBook-Pro:data-shell janakin$ cd data
[Janakins-MacBook-Pro:data janakin$ ls
amino-acids.txt  animals.txt      elements      pdb          salmon.txt
animal-counts    backup          morse.txt    planets.txt  sunspot.txt
[Janakins-MacBook-Pro:data janakin$ head -n 3 animals.txt > animals-subset.txt
[Janakins-MacBook-Pro:data janakin$ tail -n 2 animals.txt >> animals-subset.txt
Janakins-MacBook-Pro:data janakin$
```



4/9/19 - 1:11pm

Having a go at piping.

```

Last login: Wed Sep  4 13:04:33 on ttys000
[Janakins-MacBook-Pro:~ janakin$ cd Desktop/data-shell/molecules/
[Janakins-MacBook-Pro:molecules janakin$ ls
animals-subset.txt      methane.pdb          sorted-lengths.txt
cubane.pdb              octane.pdb          testfile01.txt
ethane.pdb              pentane.pdb         testfile02.txt
lengths.txt             propane.pdb
[Janakins-MacBook-Pro:molecules janakin$ sort -n lengths | head -n 1
sort: No such file or directory
[Janakins-MacBook-Pro:molecules janakin$ sort -n lengths.txt | head -n 1
  9 methane.pdb
[Janakins-MacBook-Pro:molecules janakin$ wc -l *.pdb | sort -n
  9 methane.pdb
 12 ethane.pdb
 15 propane.pdb
 20 cubane.pdb
 21 pentane.pdb
 30 octane.pdb
 107 total
[Janakins-MacBook-Pro:molecules janakin$ wc -l *.pdb | sort -n | head -n 1
  9 methane.pdb
Janakins-MacBook-Pro:molecules janakin$ 

```

4/9/19 - 1:13pm

Piping Commands Together exercise.

The answer is option 4.

```
wc -l * | sort -n | head -n 3
```

will get us 3 files which have the least number of lines.

4/9/19 - 1:15pm

Pipe Reading Comprehension exercise.

As expected, I had the same answer as the exercise.

```
2012-11-06,rabbit
2012-11-06,deer
2012-11-05,raccoon
```

4/9/19 - 1:19pm

Pipe Construction exercise.

The solution is

```
cut -d , -f 2 animals.txt | sort | uniq
```

4/9/19 - 1:21pm

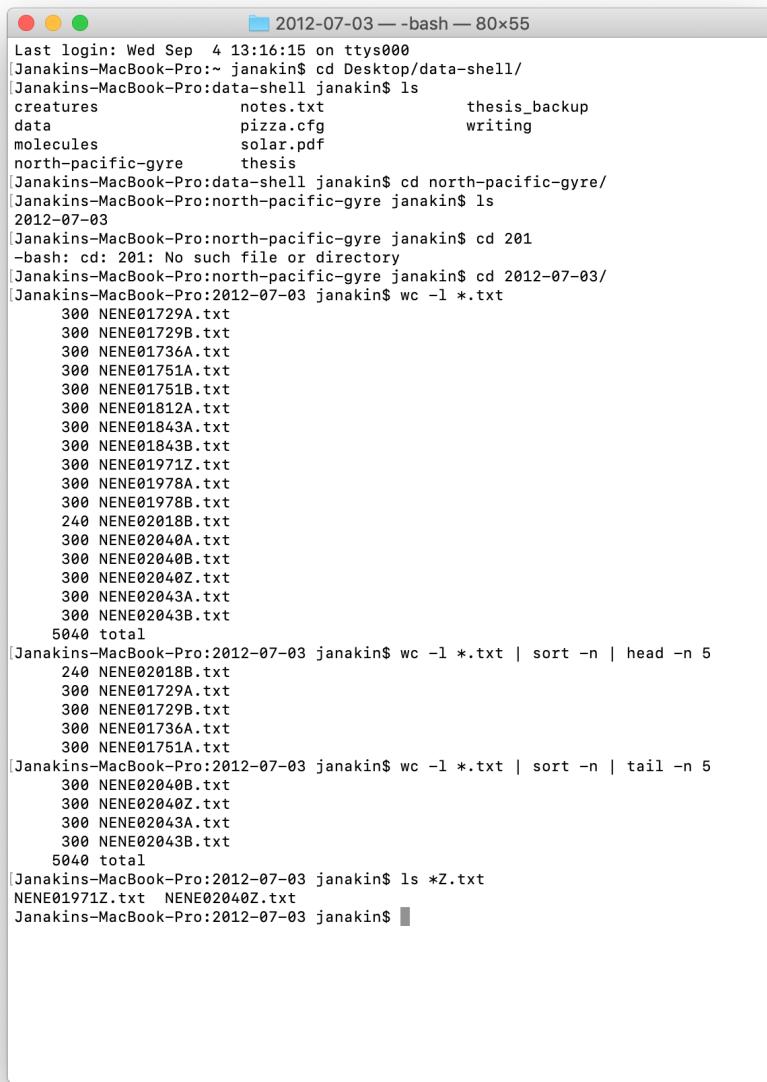
Which Pipe? exercise.

The solution is

```
cut -d, -f 2 animals.txt | sort | uniq -c | wc -l
```

4/9/19 - 1:25pm

Continuing with Nelle's examples.



```
Last login: Wed Sep  4 13:16:15 on ttys000
[Janakins-MacBook-Pro:~ janakin$ cd Desktop/data-shell/
[Janakins-MacBook-Pro:data-shell janakin$ ls
creatures          notes.txt           thesis_backup
data               pizza.cfg            writing
molecules          solar.pdf
north-pacific-gyre thesis
[Janakins-MacBook-Pro:data-shell janakin$ cd north-pacific-gyre/
[Janakins-MacBook-Pro:north-pacific-gyre janakin$ ls
2012-07-03
[Janakins-MacBook-Pro:north-pacific-gyre janakin$ cd 201
[bash: cd: 201: No such file or directory
[Janakins-MacBook-Pro:north-pacific-gyre janakin$ cd 2012-07-03/
[Janakins-MacBook-Pro:2012-07-03 janakin$ wc -l *.txt
 300 NENE01729A.txt
 300 NENE01729B.txt
 300 NENE01736A.txt
 300 NENE01751A.txt
 300 NENE01751B.txt
 300 NENE01812A.txt
 300 NENE01843A.txt
 300 NENE01843B.txt
 300 NENE01971Z.txt
 300 NENE01978A.txt
 300 NENE01978B.txt
 240 NENE02018B.txt
 300 NENE02040A.txt
 300 NENE02040B.txt
 300 NENE02040Z.txt
 300 NENE02043A.txt
 300 NENE02043B.txt
5040 total
[Janakins-MacBook-Pro:2012-07-03 janakin$ wc -l *.txt | sort -n | head -n 5
 240 NENE02018B.txt
 300 NENE01729A.txt
 300 NENE01729B.txt
 300 NENE01736A.txt
 300 NENE01751A.txt
[Janakins-MacBook-Pro:2012-07-03 janakin$ wc -l *.txt | sort -n | tail -n 5
 300 NENE02040B.txt
 300 NENE02040Z.txt
 300 NENE02043A.txt
 300 NENE02043B.txt
5040 total
[Janakins-MacBook-Pro:2012-07-03 janakin$ ls *Z.txt
NENE01971Z.txt  NENE02040Z.txt
Janakins-MacBook-Pro:2012-07-03 janakin$
```

4/9/19 - 1:27pm

Wildcard Expressions exercise.

Answers.

1.

```
ls *A.txt  
ls *B.txt
```
2. They are separated because they are two commands
3. Only when there are no files ending in A.txt or B.txt

4/9/19 - 1:31pm

Removing Unneeded Files exercise.

Only

```
rm *.txt
```

is correct.

4/9/19 - 2:08pm

On to episode 5, **Loops**. Following the instruction, I had realised that I put in the code

```
for filename in basilisk.dat minotaur.dat unicorn.dat  
> do  
>     head -n 2 $filename | tail -n 1  
> done
```

as one continuous code. I then realised that each line was a separate command input.

```

Last login: Wed Sep  4 14:01:36 on ttys000
Janakins-MacBook-Pro:~ janakin$ cd Desktop/data-shell/creatures/
Janakins-MacBook-Pro:creatures janakin$ head -n 5 basilisk.dat minotaur.dat unicorn.dat
==> basilisk.dat <==
COMMON NAME: basilisk
CLASSIFICATION: basiliscus vulgaris
UPDATED: 1745-05-02
CCCCAACGAG
GAAACAGATC

==> minotaur.dat <==
COMMON NAME: minotaur
CLASSIFICATION: bos hominus
UPDATED: 1765-02-17
CCCGAAGGAC
CGACATCTCT

==> unicorn.dat <==
COMMON NAME: unicorn
CLASSIFICATION: equus monoceros
UPDATED: 1738-11-24
GCCCGGGTCG
CTTACCTTA
Janakins-MacBook-Pro:creatures janakin$ for filename in basilisk.dat minotaur.dat unicorn.dat >
| do > head -n 2 $filename | tail -n 1 > done
| -bash: syntax error near unexpected token `>
| Janakins-MacBook-Pro:creatures janakin$ for filename in basilisk.dat minotaur.dat unicorn.dat
| > do
| > head -n 2 $filename | tail -n 1
| > done
| CLASSIFICATION: basiliscus vulgaris
| CLASSIFICATION: bos hominus
| CLASSIFICATION: equus monoceros

```

4/9/19 - 2:22pm

Completing *Variables in Loops* exercise. The output of code

```
$ for datafile in *.pdb
> do
>     ls *.pdb
> done
```

is

```
Last login: Wed Sep  4 14:03:54 on ttys000
[Janakins-MacBook-Pro:~ janakin$ cd Desktop/data-shell/molecules/
[Janakins-MacBook-Pro:molecules janakin$ ls
animals-subset.txt      methane.pdb          sorted-lengths.txt
cubane.pdb               octane.pdb          testfile01.txt
ethane.pdb               pentane.pdb         testfile02.txt
lengths.txt              propane.pdb
[Janakins-MacBook-Pro:molecules janakin$ for datafile in *.pdb
[> do
[> ls *.pdb
[> done
cubane.pdb      methane.pdb      pentane.pdb
ethane.pdb       octane.pdb      propane.pdb
Janakins-MacBook-Pro:molecules janakin$
```

The output code of

```
$ for datafile in *.pdb
> do
>ls $datafile
> done
```

is

```

|> done
cubane.pdb      methane.pdb      pentane.pdb
ethane.pdb      octane.pdb      propane.pdb
[Janakins-MacBook-Pro:molecules janakin$ for datafile in *.pdb
|> do
|> ls $datafile
|> done
cubane.pdb
ethane.pdb
methane.pdb
octane.pdb
pentane.pdb
propane.pdb
[Janakins-MacBook-Pro:molecules janakin$
```

The second code lists a different file on each loop iteration.

4/9/19 - 2:28pm

Completing *Limiting Sets of Files* exercise.

```
$ for filename in c*
> do
>   ls $filename
> done
```

lists only cubane.pdb.

```
$ for filename in *c*
> do
>   ls $filename
> done
```

lists both cubane.pdb and octane.pdb because of the presence of the c character.

4/9/19 - 2:32pm

Completing *Saving to a File in a Loop - Part One* exercise.

```
for alkanes in *.pdb
do
echo $alkanes
cat $alkanes > alkanes.pdb
done
```

has text from each file written into alkanes.pdb. Each iteration overwrites the last. The propane.pdb is the last to be written in alkanes.pdb.

4/9/19 - 2:36pm

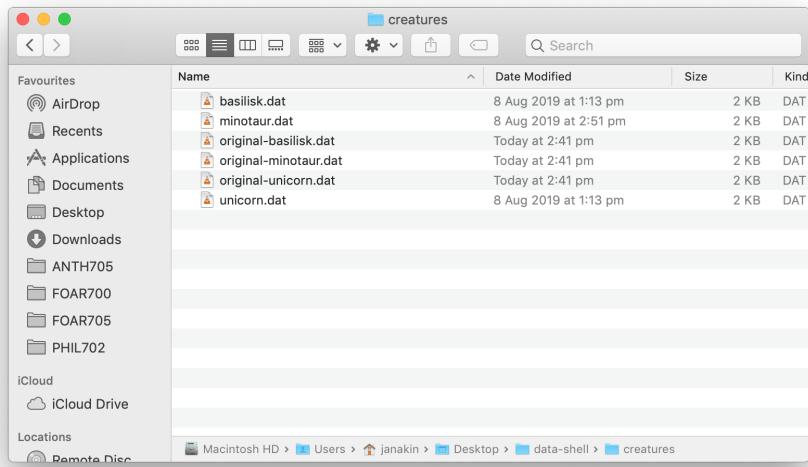
Completing *Saving to a File in a Loop - Part Two* exercise.

All of the text from cubane.pdb, ethane.pdb, methane.pdb, octane.pdb, pentane.pdb and propane.pdb would be concatenated and saved to a file called all.pdb.

4/9/19 - 2:44pm

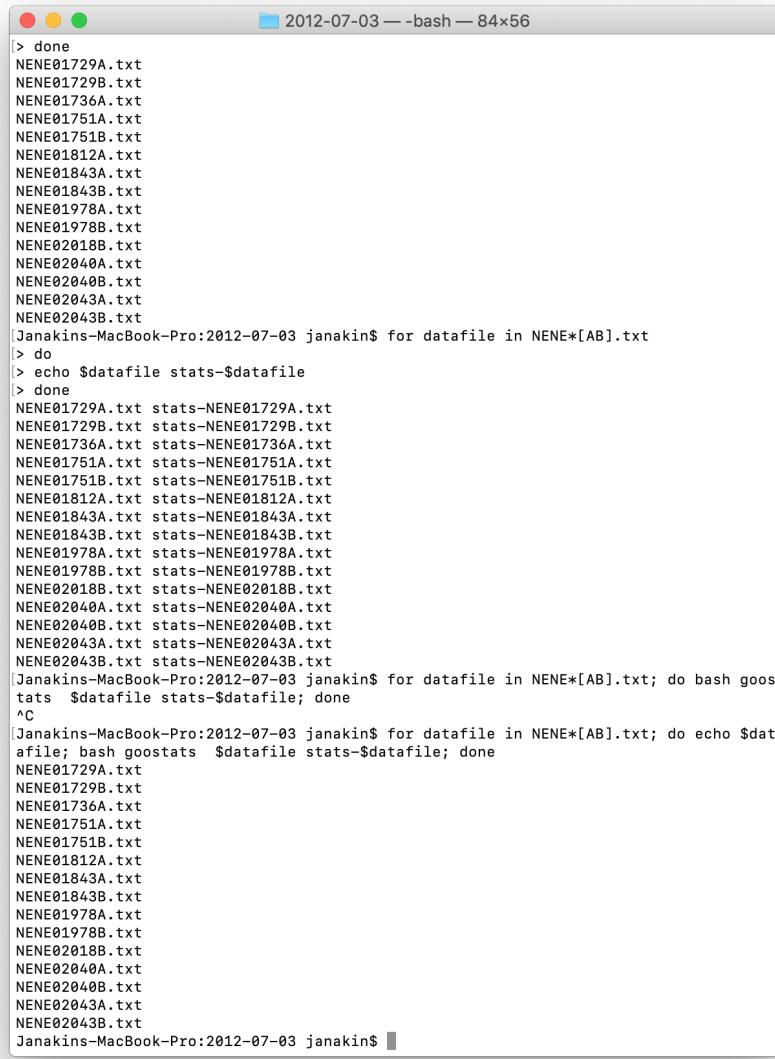
Continuing with the lesson.

```
creatures — bash — 68x14
GCGCGGGCGG
CACTCTATGT
TTTCGACAC
TGGACTGCTT
CCCTTGAGG
GTGGATTTT
CGTAACGGT
[Janakins-MacBook-Pro:creatures janakin$ pwd
/Users/janakin/Desktop/data-shell/creatures
[Janakins-MacBook-Pro:creatures janakin$ for filename in *.dat
[> do
[> cp $filename original-$filename
[> done
Janakins-MacBook-Pro:creatures janakin$
```



4/9/19 - 2:53pm

Back with Nelle's example.



```
[> done
NENE01729A.txt
NENE01729B.txt
NENE01736A.txt
NENE01751A.txt
NENE01751B.txt
NENE01812A.txt
NENE01843A.txt
NENE01843B.txt
NENE01978A.txt
NENE01978B.txt
NENE02018B.txt
NENE02040A.txt
NENE02040B.txt
NENE02043A.txt
NENE02043B.txt
[Janakins-MacBook-Pro:2012-07-03 janakin$ for datafile in NENE*[AB].txt
[> do
[> echo $datafile stats-$datafile
[> done
NENE01729A.txt stats-NENE01729A.txt
NENE01729B.txt stats-NENE01729B.txt
NENE01736A.txt stats-NENE01736A.txt
NENE01751A.txt stats-NENE01751A.txt
NENE01751B.txt stats-NENE01751B.txt
NENE01812A.txt stats-NENE01812A.txt
NENE01843A.txt stats-NENE01843A.txt
NENE01843B.txt stats-NENE01843B.txt
NENE01978A.txt stats-NENE01978A.txt
NENE01978B.txt stats-NENE01978B.txt
NENE02018B.txt stats-NENE02018B.txt
NENE02040A.txt stats-NENE02040A.txt
NENE02040B.txt stats-NENE02040B.txt
NENE02043A.txt stats-NENE02043A.txt
NENE02043B.txt stats-NENE02043B.txt
[Janakins-MacBook-Pro:2012-07-03 janakin$ for datafile in NENE*[AB].txt; do bash goos]
tats $datafile stats-$datafile; done
^C
[Janakins-MacBook-Pro:2012-07-03 janakin$ for datafile in NENE*[AB].txt; do echo $dat
afile; bash goostats $datafile stats-$datafile; done
NENE01729A.txt
NENE01729B.txt
NENE01736A.txt
NENE01751A.txt
NENE01751B.txt
NENE01812A.txt
NENE01843A.txt
NENE01843B.txt
NENE01978A.txt
NENE01978B.txt
NENE02018B.txt
NENE02040A.txt
NENE02040B.txt
NENE02043A.txt
NENE02043B.txt
Janakins-MacBook-Pro:2012-07-03 janakin$
```

Not exactly sure what bash goostats did, but it did certainly look like Terminal was doing something.

4/9/19 - 2:59pm

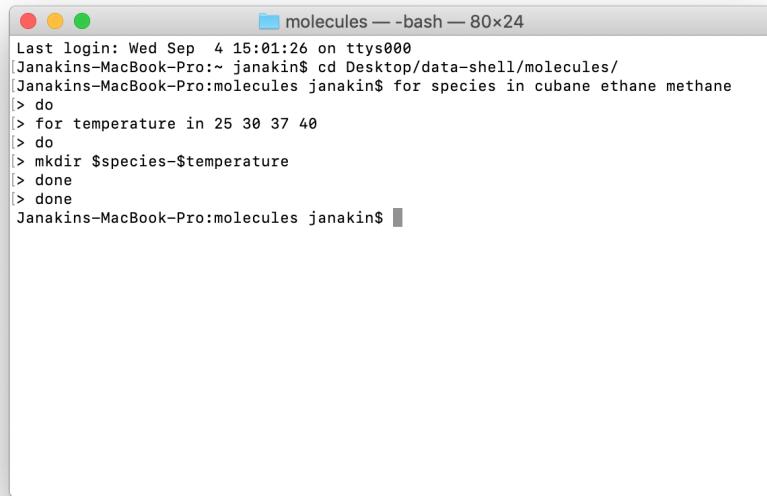
Completing *Doing a Dry Run* exercise.

The second version is the one we want to run because it prints everything.

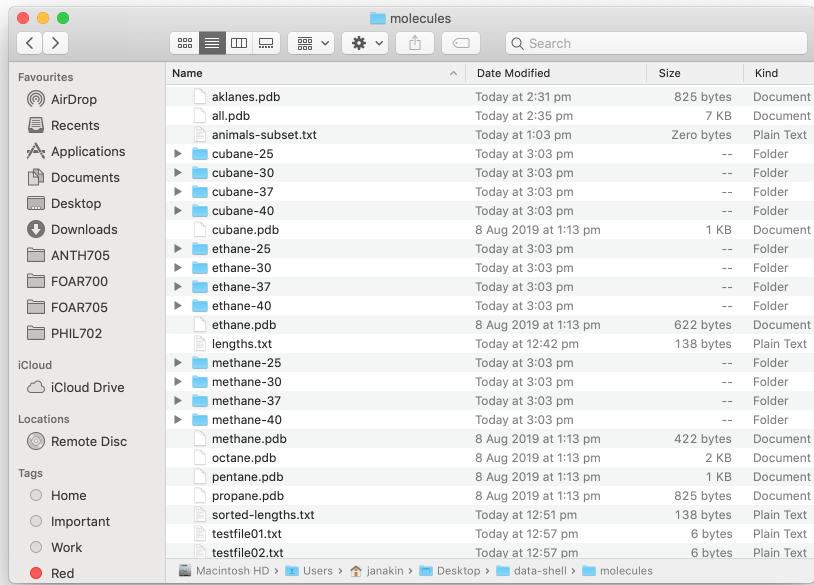
4/9/19 - 3:03pm

Completing *Nested Loops* exercise.

Not sure if this was successful.



```
Last login: Wed Sep  4 15:01:26 on ttys000
[Janakins-MacBook-Pro:~ janakin$ cd Desktop/data-shell/molecules/
[Janakins-MacBook-Pro:molecules janakin$ for species in cubane ethane methane
[> do
[> for temperature in 25 30 37 40
[> do
[> mkdir ${species}-${temperature}
[> done
[> done
Janakins-MacBook-Pro:molecules janakin$ ]]
```



4/9/19 - 3:16pm

Episode 6: Shell Scripts

Following the instructions. Below are the screenshots of what I have done.

Creating the middle.sh shell.



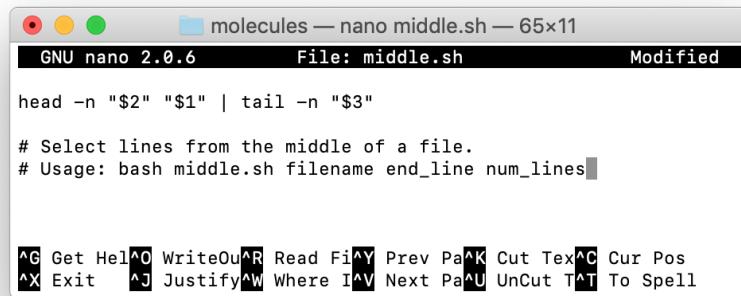
molecules — nano middle.sh — 80x24

GNU nano 2.0.6 File: middle.sh Modified

```
head -n 15 octane.pdb | tail -n 5
```

^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos
^X Exit ^J Justify ^W Where Is ^V Next Page ^U UnCut Text ^T To Spell

Recoded the middle.sh file to be more versatile and include information to other users about what it does.



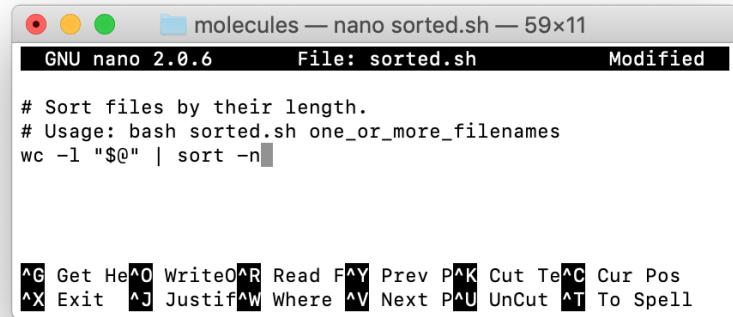
molecules — nano middle.sh — 65x11

GNU nano 2.0.6 File: middle.sh Modified

```
head -n "$2" "$1" | tail -n "$3"  
  
# Select lines from the middle of a file.  
# Usage: bash middle.sh filename end_line num_lines
```

^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos
^X Exit ^J Justify ^W Where Is ^V Next Page ^U UnCut Text ^T To Spell

Created the sorted.sh shell.



The screenshot shows a macOS terminal window titled "molecules — nano sorted.sh — 59x11". The title bar includes standard OS X window controls (red, yellow, green) and the file path. Below the title bar, the status bar displays "GNU nano 2.0.6", "File: sorted.sh", and "Modified". The main editor area contains the following text:

```
# Sort files by their length.  
# Usage: bash sorted.sh one_or_more_filenames  
wc -l "$@" | sort -n
```

At the bottom of the editor, a series of keyboard shortcuts are listed:

^G Get Help **^O** Write **^R** Read **F^Y** Prev **P^K** Cut **T^C** Cur Pos
^X Exit **^J** Justif **^W** Where **^V** Next **P^U** UnCut **^T** To Spell

Terminal window and commands used up until this point.

```

molecules -- bash -- 74x50
ATOM    13 H      1     -3.172  -1.337   0.206  1.00  0.00
[Janakins-MacBook-Pro:molecules janakin$ bash middle.sh pentane.pdb
ATOM    9 H      1      1.324   0.350  -1.332  1.00  0.00
ATOM   10 H      1      1.271   1.378   0.122  1.00  0.00
ATOM   11 H      1     -0.074  -0.384   1.288  1.00  0.00
ATOM   12 H      1     -0.048  -1.362  -0.205  1.00  0.00
ATOM   13 H      1     -1.183   0.500  -1.412  1.00  0.00
[Janakins-MacBook-Pro:molecules janakin$ nano middle.sh
[Janakins-MacBook-Pro:molecules janakin$ nano middle.sh
[Janakins-MacBook-Pro:molecules janakin$ bash middle.sh pentane.pdb 15 5
ATOM    9 H      1      1.324   0.350  -1.332  1.00  0.00
ATOM   10 H      1      1.271   1.378   0.122  1.00  0.00
ATOM   11 H      1     -0.074  -0.384   1.288  1.00  0.00
ATOM   12 H      1     -0.048  -1.362  -0.205  1.00  0.00
ATOM   13 H      1     -1.183   0.500  -1.412  1.00  0.00
[Janakins-MacBook-Pro:molecules janakin$ bash middle.sh pentane.pdb 20 5
ATOM   14 H      1     -1.259   1.420   0.112  1.00  0.00
ATOM   15 H      1     -2.608  -0.407   1.130  1.00  0.00
ATOM   16 H      1     -2.540  -1.303  -0.404  1.00  0.00
ATOM   17 H      1     -3.393   0.254  -0.321  1.00  0.00
TER    18          1
[Janakins-MacBook-Pro:molecules janakin$ nano middle.sh

Janakins-MacBook-Pro:molecules janakin$ nano sorted.sh
[Janakins-MacBook-Pro:molecules janakin$ bash sorted.sh *pdb ../creatures/*]
.dat
    9 methane.pdb
    12 ethane.pdb
    15 aklanes.pdb
    15 propane.pdb
    20 cubane.pdb
    21 pentane.pdb
    30 octane.pdb
    122 all.pdb
    163 ../creatures/basilisk.dat
    163 ../creatures/minotaur.dat
    163 ../creatures/original-basilisk.dat
    163 ../creatures/original-minotaur.dat
    163 ../creatures/original-unicorn.dat
    163 ../creatures/unicorn.dat
    1222 total
Janakins-MacBook-Pro:molecules janakin$
```

4/9/19 - 3:34pm

List Unique Species exercise. The script I would use.

```

for file in $@
do
echo "Unique species in $file:"
# Extract species names
cut -d , -f 2 $file | sort | uniq
done

```

4/9/19 - 3:36pm

Why Record Commands in the History Before Running Them

It is a good idea to do this because it gives you a record of what commands you used, and in what order. So if things go wrong, you can trouble shoot.

4/9/19 - 3:41pm

Following on with Nelle's example.



The screenshot shows a terminal window titled '2012-07-03 — nano do-stats.sh — 80x24'. The window title bar includes 'File: do-stats.sh'. The main area of the window displays the following shell script:

```

# Calculate stats for data files.
for datafile in "$@"
do
    echo $datafile
    bash goostats $datafile stats-$datafile
done

```

At the bottom of the window, there is a status bar with the text '[Read 6 lines]'. Below the status bar, a series of keyboard shortcuts are listed:

- [G] Get Help
- [W] WriteOut
- [R] Read File
- [Y] Prev Page
- [K] Cut Text
- [C] Cur Pos
- [X] Exit
- [J] Justify
- [W] Where Is
- [V] Next Page
- [U] UnCut Text
- [T] To Spell

The screenshot shows a terminal window titled "2012-07-03 — nano do-stats.sh — 80x24". The file "do-stats.sh" contains the following code:

```
# Calculate stats for data files.
for datafile in "$@"
do
    echo $datafile
    bash goostats $datafile stats-$datafile
done
```

At the bottom of the screen, there is a status bar with the text "[Read 6 lines]" and a series of keyboard shortcuts:

- ^G Get Help
- ^O WriteOut
- ^R Read File
- ^Y Prev Page
- ^K Cut Text
- ^C Cur Pos
- ^X Exit
- ^J Justify
- ^W Where Is
- ^V Next Page
- ^U UnCut Text
- ^T To Spell

4/9/19 - 3:43pm

Variables in Shell Scripts

The correct answer is 2.The first and the last line of each file ending in .pdb in the molecules directory

4/9/19 - 3:44pm

Find the Longest File With a Given Extension

I would use the code

```
wc -l $1/*.\$2 | sort -n | tail -n 2 | head -n 1
```

4/9/19 - 3:47pm

Script Reading Comprehension

Script 1 would print out a list of all files containing a dot. Script 2 would print the contents of the first 3 files with a .pdb extension. Script 3 would print all the arguments to the script followed by .pdb.

4/9/19 - 3:48pm

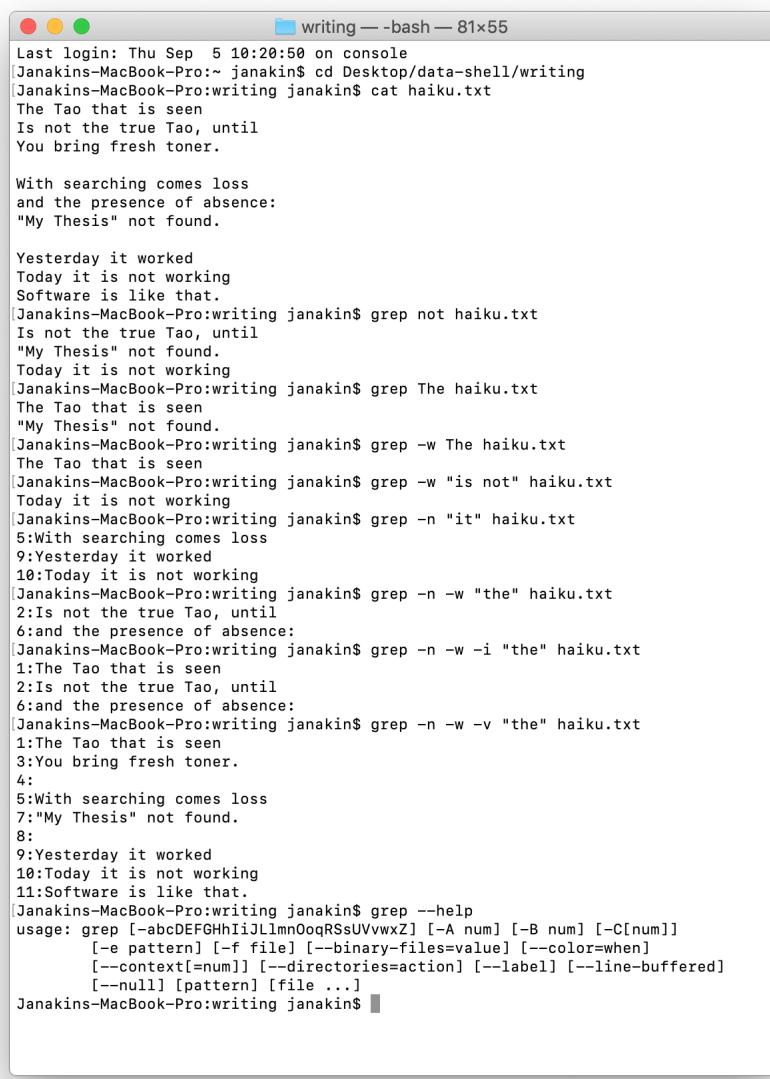
Debugging Script

The -x option gets bash to run in debug mode, printing out each command as it runs it, which will help when trouble shooting.

4/9/19 - 10:44am

Am now on the 6th and final episode **Finding Things**.

The first little bit.



```
Last login: Thu Sep  5 10:20:50 on console
[Janakins-MacBook-Pro:~ janakin$ cd Desktop/data-shell/writing
[Janakins-MacBook-Pro:writing janakin$ cat haiku.txt
The Tao that is seen
Is not the true Tao, until
You bring fresh toner.

With searching comes loss
and the presence of absence:
"My Thesis" not found.

Yesterday it worked
Today it is not working
Software is like that.
[Janakins-MacBook-Pro:writing janakin$ grep not haiku.txt
Is not the true Tao, until
"My Thesis" not found.
Today it is not working
[Janakins-MacBook-Pro:writing janakin$ grep The haiku.txt
The Tao that is seen
"My Thesis" not found.
[Janakins-MacBook-Pro:writing janakin$ grep -w The haiku.txt
The Tao that is seen
[Janakins-MacBook-Pro:writing janakin$ grep -w "is not" haiku.txt
Today it is not working
[Janakins-MacBook-Pro:writing janakin$ grep -n "it" haiku.txt
5:With searching comes loss
9:Yesterday it worked
10:Today it is not working
[Janakins-MacBook-Pro:writing janakin$ grep -n -w "the" haiku.txt
2:Is not the true Tao, until
6:and the presence of absence:
[Janakins-MacBook-Pro:writing janakin$ grep -n -w -i "the" haiku.txt
1:The Tao that is seen
2:You bring fresh toner.
4:
5:With searching comes loss
7:"My Thesis" not found.
8:
9:Yesterday it worked
10:Today it is not working
11:Software is like that.
[Janakins-MacBook-Pro:writing janakin$ grep --help
usage: grep [-abcDEFGHHiijJlmnOoqRSsUVvwxZ] [-A num] [-B num] [-C[num]]
          [-e pattern] [-f file] [--binary-files=value] [--color=when]
          [--context[=num]] [--directories=action] [--label] [--line-buffered]
          [--null] [pattern] [file ...]
Janakins-MacBook-Pro:writing janakin$
```

4/9/19 - 10:50am

Using grep

The correct answer is 3

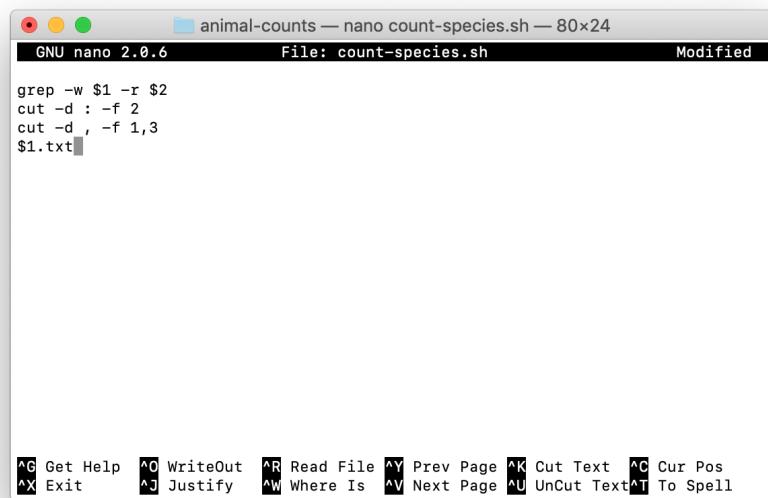
```
grep -w "of" haiku.txt
```

because it will only look for whole words.

4/9/19 - 10:57am

Tracking a Species

I made this shell script.



A screenshot of a terminal window titled "animal-counts — nano count-species.sh — 80x24". The window shows the following shell script:

```
GNU nano 2.0.6           File: count-species.sh           Modified
grep -w $1 -r $2
cut -d : -f 2
cut -d , -f 1,3
$1.txt
```

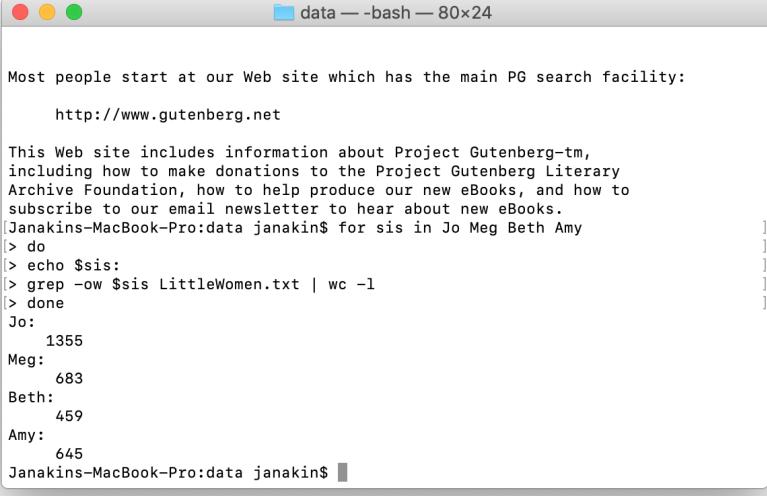
At the bottom of the window, there is a menu bar with various keyboard shortcuts:

**^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos
^X Exit ^J Justify ^W Where Is ^V Next Page ^U UnCut Text ^T To Spell**

4/9/19 - 11:00am

Little Women

I did the following.



Most people start at our Web site which has the main PG search facility:
<http://www.gutenberg.net>

This Web site includes information about Project Gutenberg-tm,
including how to make donations to the Project Gutenberg Literary
Archive Foundation, how to help produce our new eBooks, and how to
subscribe to our email newsletter to hear about new eBooks.

```
[Janakins-MacBook-Pro:~/data] janakin$ for sis in Jo Meg Beth Amy
|> do
|> echo $sis:
|> grep -ow $sis LittleWomen.txt | wc -l
|> done
Jo:
    1355
Meg:
    683
Beth:
    459
Amy:
    645
[Janakins-MacBook-Pro:~/data] janakin$
```

4/9/19 - 11:04am

Continuing on with the lesson.

```
writing — bash — 69x42
Last login: Thu Sep  5 10:59:28 on ttys000
[Janakins-MacBook-Pro:~ janakin$ cd Desktop/data-shell/writing/
[Janakins-MacBook-Pro:writing janakin$ find .
.
./tools
./tools/old
./tools/old/oldtool
./tools/format
./tools/stats
./haiku.txt
./thesis
./thesis/empty-draft.md
./data
./data/two.txt
./data/LittleWomen.txt
./data/one.txt
[Janakins-MacBook-Pro:writing janakin$ find . -type d
.
./tools
./tools/old
./thesis
./data
[Janakins-MacBook-Pro:writing janakin$ find . -type f
./tools/old/oldtool
./tools/format
./tools/stats
./haiku.txt
./thesis/empty-draft.md
./data/two.txt
./data/LittleWomen.txt
./data/one.txt
[Janakins-MacBook-Pro:writing janakin$ find . -name *.txt
./haiku.txt
[Janakins-MacBook-Pro:writing janakin$ find . -name '*.txt'
./haiku.txt
./data/two.txt
./data/LittleWomen.txt
./data/one.txt
Janakins-MacBook-Pro:writing janakin$ ]
```

4/9/19 - 11:06am

Matching and Subtracting

The correct answer is 1.

```
find data -name '*s.txt' | grep -v net
```

4/9/19 - 11:10am

find Pipeline Reading Comprehension

It will find .dat files, count the number of lines the files have and sort the output numerically.

4/9/19 - 11:12am

Finding Files With Different Properties

```
find ./ -type f -mtime -1 -user ahmed
```