Lecture 04 - Unix loops

for loops in general

for variable in set

code to be executed

- variable: a place holder used to temporarily hold each element in the set
- set: the entities you want to have the code within the for loop executed for

for loops in Unix

```
for variable in set
do
    code to be executed using $variable
done
```

potato example

```
for number in one two
do
echo $number potato
done
```

```
## one potato
## two potato
```

Executing a for loop at the Unix command line

- usually the Unix command prompt is a \$
- once you've begun a for loop the prompt changes to >
- this is only to remind you that you are in a loop
- you can get out of the loop or any running command with Ctrl-c

Reminder again

Most often the variables we will include in a set are file names, therefore you should remember. . .

▶ NO SPACES IN FILE NAMES!!!!!

File extensions don't mean anything to Unix!

By the way...

Did anyone else get files with name stats-stats-filename when working through tutorial?

Why?

How to avoid this?

More than one shell window!!!

Holy cows, you can do this?

Remember multi-processing!

Echo and debugging

Why is echo useful for testing for loops?

Echo and debugging

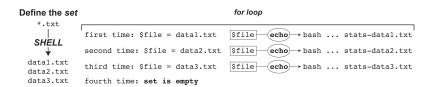
Why is echo useful for testing for loops?

```
for file in *.txt
do
   echo "bash goostats $file stats-$file"
done
```

```
## bash goostats data1.txt stats-data1.txt
## bash goostats data2.txt stats-data2.txt
## bash goostats data3.txt stats-data3.txt
```

What happened on a process level?

```
for file in *.txt
do
   echo "bash goostats $file stats-$file"
done
```



for loop application

Challenge: Use the files in molecules/ to print each # of atoms:

cubane:
16
ethane:
8
methane:

5
octane:
26

pentane:

17 propane:

11

for loop application

```
for file in *.pdb
do
  echo $file | sed 's/.pdb/: /'
  cat $file | grep ATOM | wc -l
done
```

storing for loop output in a file

```
for file in *.pdb
do
   echo $file | sed 's/.pdb/: /' > output.txt
   cat $file | grep ATOM | wc -l > output.txt
done
```

VS

```
for file in *.pdb
do
  echo $file | sed 's/.pdb/: /' >> output.txt
  cat $file | grep ATOM | wc -l >> output.txt
done
```

nested for loops

As shown in the tutorial, this is useful if you have multiple directories with similar files

```
for dir in */
do
  for file in *.txt
  do
    echo "$file is in $dir"
  done
done
```

function review

```
for

$

history

up arrow

Ctrl-c

Ctrl-a

Ctrl-e
```