The Unix Shell

Pipes and Filters
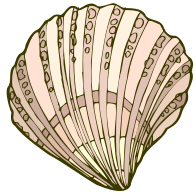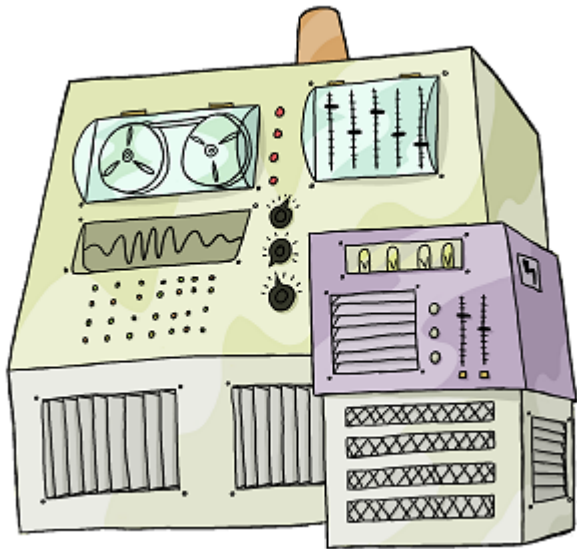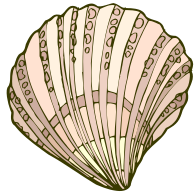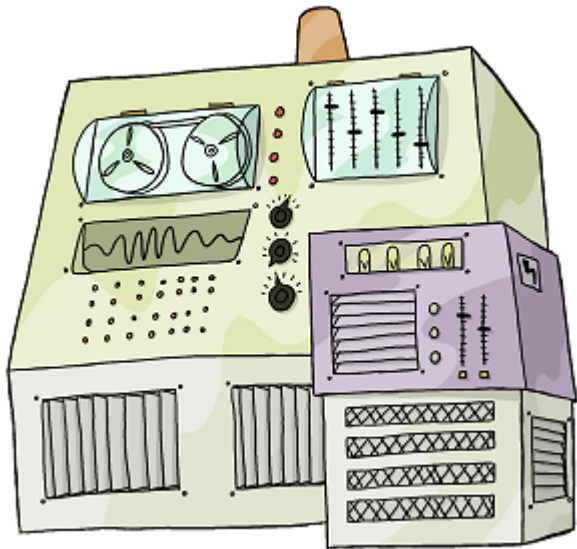
shell

shell

pwd	mkdir
cd	nano
ls	rm
.	rmdir
..	mv
	cp
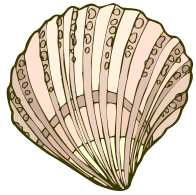
shell
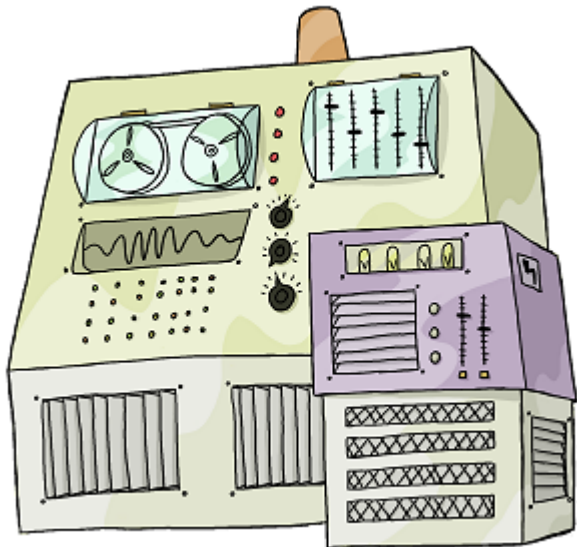
pwd	mkdir
cd	nano
ls	rm
.	rmdir
..	mv
	cp

More powerful

when combined

```
$ ls molecules
```

cubane.pdb   *ethane.pdb*   *methane.pdb*

octane.pdb   *pentane.pdb*   *propane.pdb*

$

```
$ ls molecules

cubane.pdb     ethane.pdb     methane.pdb
octane.pdb     pentane.pdb    propane.pdb
$ cd molecules
$
```

```
$ ls molecules
```

cubane.pdb   *ethane.pdb*   *methane.pdb*

octane.pdb   *pentane.pdb*   *propane.pdb*

```
$ cd molecules
```

```
$ wc *.pdb
```

```
$ ls molecules
cubane.pdb      ethane.pdb      methane.pdb
octane.pdb      pentane.pdb     propane.pdb
$ cd molecules
$ wc *.pdb
```

← * is a *wild card*

```
$ ls molecules
```
cubane.pdb   *ethane.pdb*   *methane.pdb*

octane.pdb   *pentane.pdb*   *propane.pdb*
```
$ cd molecules
```
```
$ wc *.pdb
```
⟵ * is a *wild card*

matches zero or more characters

```
$ ls molecules
cubane.pdb      ethane.pdb      methane.pdb
octane.pdb      pentane.pdb     propane.pdb
$ cd molecules
$ wc *.pdb
```

* is a *wild card*

matches zero or more characters

so `*.pdb` matches all filenames

ending in `.pdb`

```
$ ls molecules
```

cubane.pdb   *ethane.pdb*   *methane.pdb*

octane.pdb   *pentane.pdb*   *propane.pdb*

```
$ cd molecules
$ wc *.pdb
```

wc cubane.pdb ethane.pdb methane.pdb octane.pdb pentane.pdb p...

```
$ ls molecules
cubane.pdb      ethane.pdb      methane.pdb
octane.pdb      pentane.pdb     propane.pdb
$ cd molecules
$ wc *.pdb
```
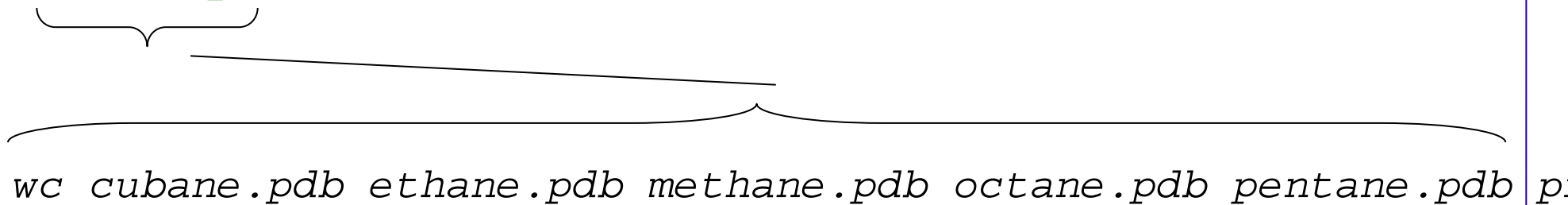← word count

```
$ ls molecules
cubane.pdb      ethane.pdb      methane.pdb
octane.pdb      pentane.pdb     propane.pdb
$ cd molecules
$ wc *.pdb
```

← word count

counts lines, words, and characters in files

```
$ ls molecules
cubane.pdb       ethane.pdb      methane.pdb
octane.pdb       pentane.pdb     propane.pdb
$ cd molecules
$ 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
$
```

```
$ wc -l *.pdb                    report only lines
   20   cubane.pdb
   12   ethane.pdb
    9   methane.pdb
   30   octane.pdb
   21   pentane.pdb
   15   propane.pdb
  107   total
$
```

```
$ wc -l *.pdb                    report only lines
   20   cubane.pdb               use -w for words or
   12   ethane.pdb               -c for characters
    9   methane.pdb
   30   octane.pdb
   21   pentane.pdb
   15   propane.pdb
  107   total
$
```

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

Which file is shortest?

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

Which file is shortest?

Easy to see when there are six…

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

Which file is shortest?

Easy to see when there are six…

…but what if there were 6000?
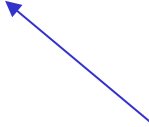
```
$ wc -l *.pdb > lengths
$
```

```
$ wc -l *.pdb > lengths
$
```
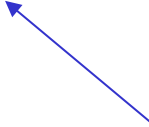
redirect output to a file

```
$ wc -l *.pdb > lengths
$
```

redirect output to a file

create file if it doesn't exist

```
$ wc -l *.pdb > lengths
$
```

redirect output to a file

create file if it doesn't exist

overwrite it if it does

```
$ wc -l *.pdb > lengths
$
```

no screen output

```
$ wc -l *.pdb > lengths
$ ls lengths
lengths
$
```

```
$ wc -l *.pdb > lengths
$ ls lengths
lengths
$ cat lengths
  20   cubane.pdb
  12   ethane.pdb
   9   methane.pdb
  30   octane.pdb
  21   pentane.pdb
  15   propane.pdb
 107   total
$
```

```
$ wc -l *.pdb > lengths
$ ls lengths
lengths
$ cat lengths          ← concatenate files
   20   cubane.pdb
   12   ethane.pdb
    9   methane.pdb
   30   octane.pdb
   21   pentane.pdb
   15   propane.pdb
  107   total
$
```

```
$ wc -l *.pdb > lengths
$ ls lengths
lengths
$ cat lengths
   20   cubane.pdb
   12   ethane.pdb
    9   methane.pdb
   30   octane.pdb
   21   pentane.pdb
   15   propane.pdb
  107   total
$
```

concatenate files

in this case, only one

so file contents printed to screen

```
$ sort -n lengths
    9   methane.pdb
   12   ethane.pdb
   15   propane.pdb
   20   cubane.pdb
   21   pentane.pdb
   30   octane.pdb
  107   total
$
```

```
$ sort -n lengths > sorted-lengths
$
```

```
$ sort -n lengths > sorted-lengths
$ head -1 sorted-lengths
    9   methane.pdb
$
```

```
$ sort -n lengths > sorted-lengths
$ head -1 sorted-lengths
    9   methane.pdb
$
```

get the first line of the file

```
$ sort -n lengths > sorted-lengths
$ head -1 sorted-lengths
    9   methane.pdb
$
```

get the first line of the file

this must be the PDB file

with the fewest lines,

since `sorted-lengths` holds

files and line counts in

order from least to greatest

```
$ sort -n lengths > sorted-lengths
$ head -1 sorted-lengths
    9   methane.pdb
$
```

get the first line of the file

this must be the PDB file

with the fewest lines,

since sorted-lengths holds

files and line counts in

order from least to greatest

not particularly obvious

```
$ sort -n lengths | head -1
```
   9   methane.pdb

$

```
$ sort -n lengths | head -1
  9   methane.pdb
$
```

a *pipe*

```
$ sort -n lengths | head -1
   9   methane.pdb
$
```

a *pipe*

use output of left side

```
$ sort -n lengths | head -1
  9  methane.pdb
$
```

a *pipe*

use output of left side

as input to right side

```
$ sort -n lengths | head -1
  9   methane.pdb
$
```

a *pipe*

use output of left side

as input to right side

without creating temporary file

```
$ wc -l *.pdb | sort -n | head -1
    9  methane.pdb
$
```

don't need to create `lengths` file

```
$ wc -l *.pdb | sort -n | head -1
  9  methane.pdb
$
```

This simple idea is why Unix has been so successful

```
$ wc -l *.pdb | sort -n | head -1
  9   methane.pdb
$
```

This simple idea is why Unix has been so successful

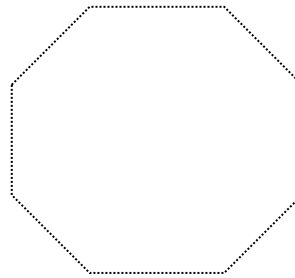## Create simple tools that:

```
$ wc -l *.pdb | sort -n | head -1
   9   methane.pdb
$
```
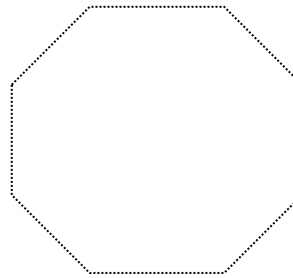
This simple idea is why Unix has been so successful

Create simple tools that:

– do one job well

```
$ wc -l *.pdb | sort -n | head -1
   9  methane.pdb
$
```

This simple idea is why Unix has been so successful

Create simple tools that:

– do one job well

– work well with each other

```
$ wc -l *.pdb | sort -n | head -1
   9   methane.pdb
$
```

This simple idea is why Unix has been so successful

Create simple tools that:
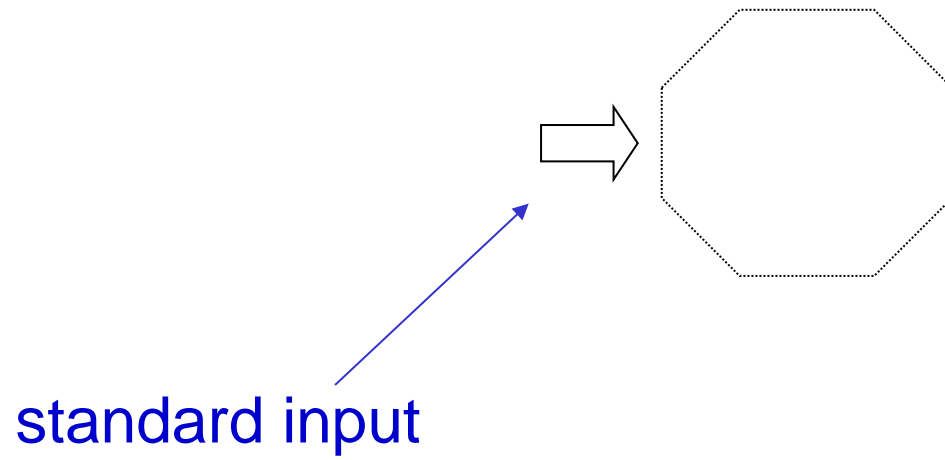
– do one job well
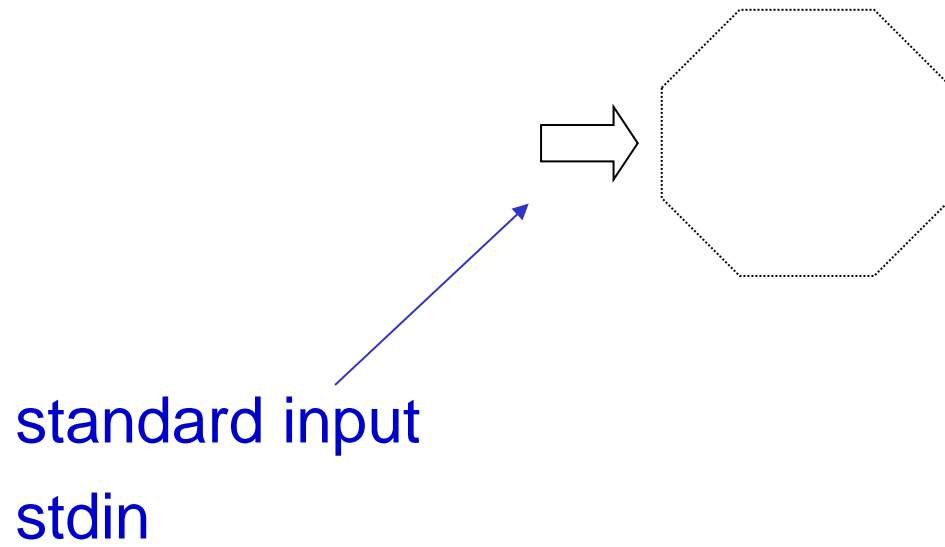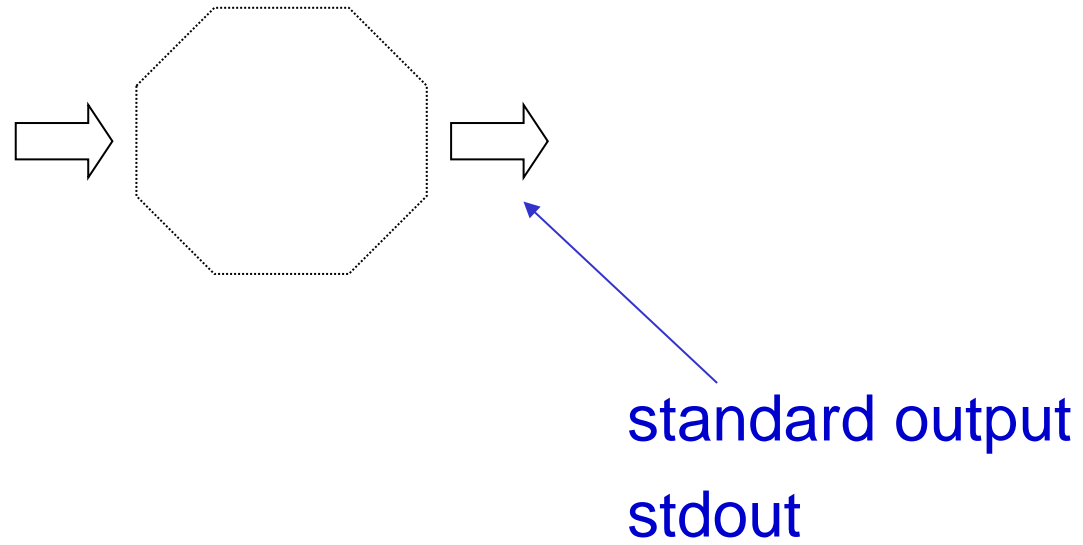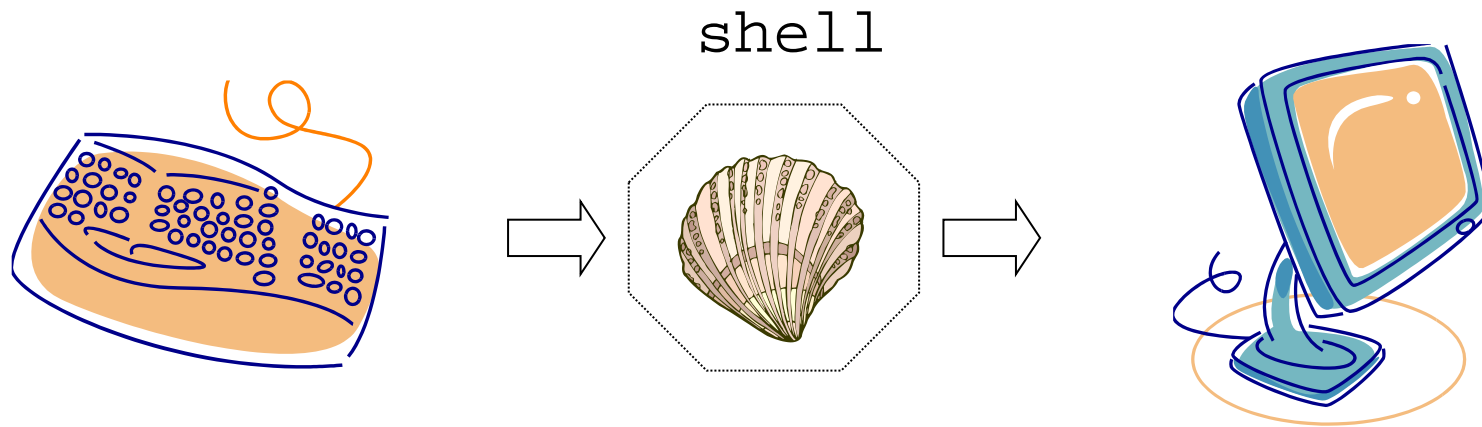
– work well with each other

10 tools can be combined in 100 ways

running program

running program

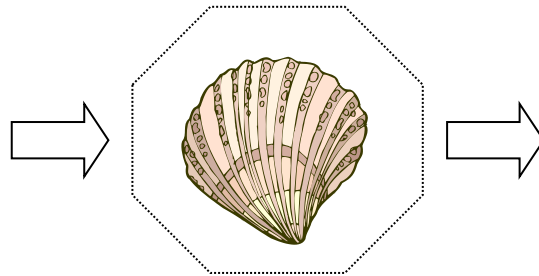*process*

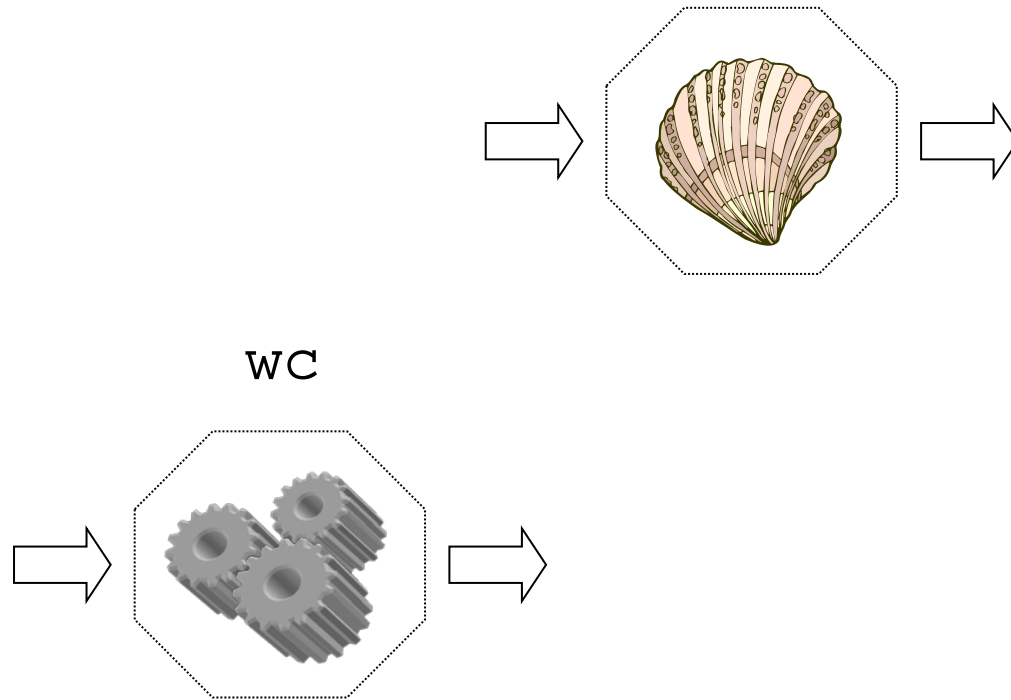standard input

standard input
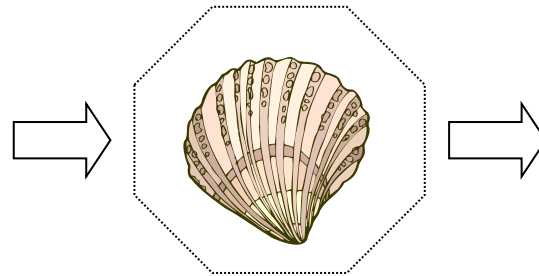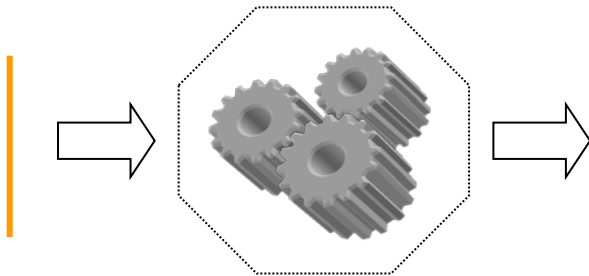
stdin

standard output

stdout

shell

```
$ wc -l *.pdb > lengths
```

`wc`
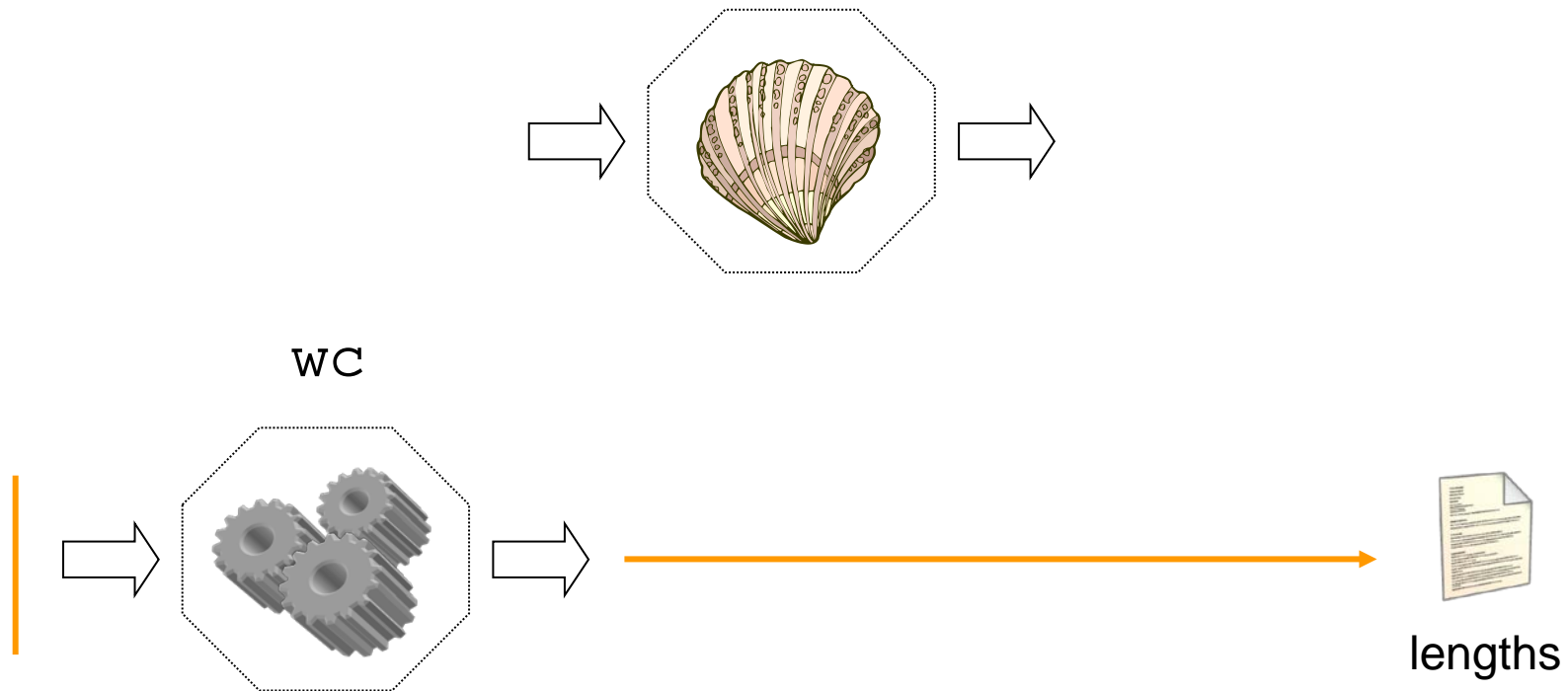
```
$ wc -l *.pdb > lengths
```

wc

$ wc -l *.pdb > lengths

wc

lengths
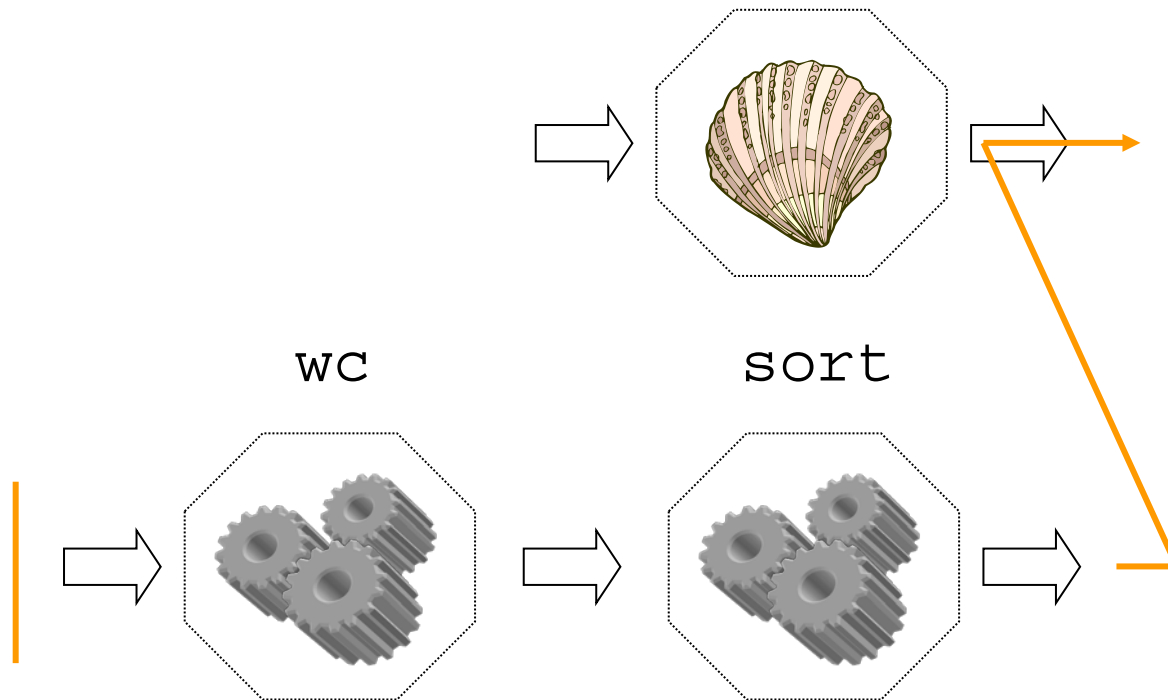
```
$ wc -l *.pdb > lengths
```

```
$ wc -l *.pdb | sort
```
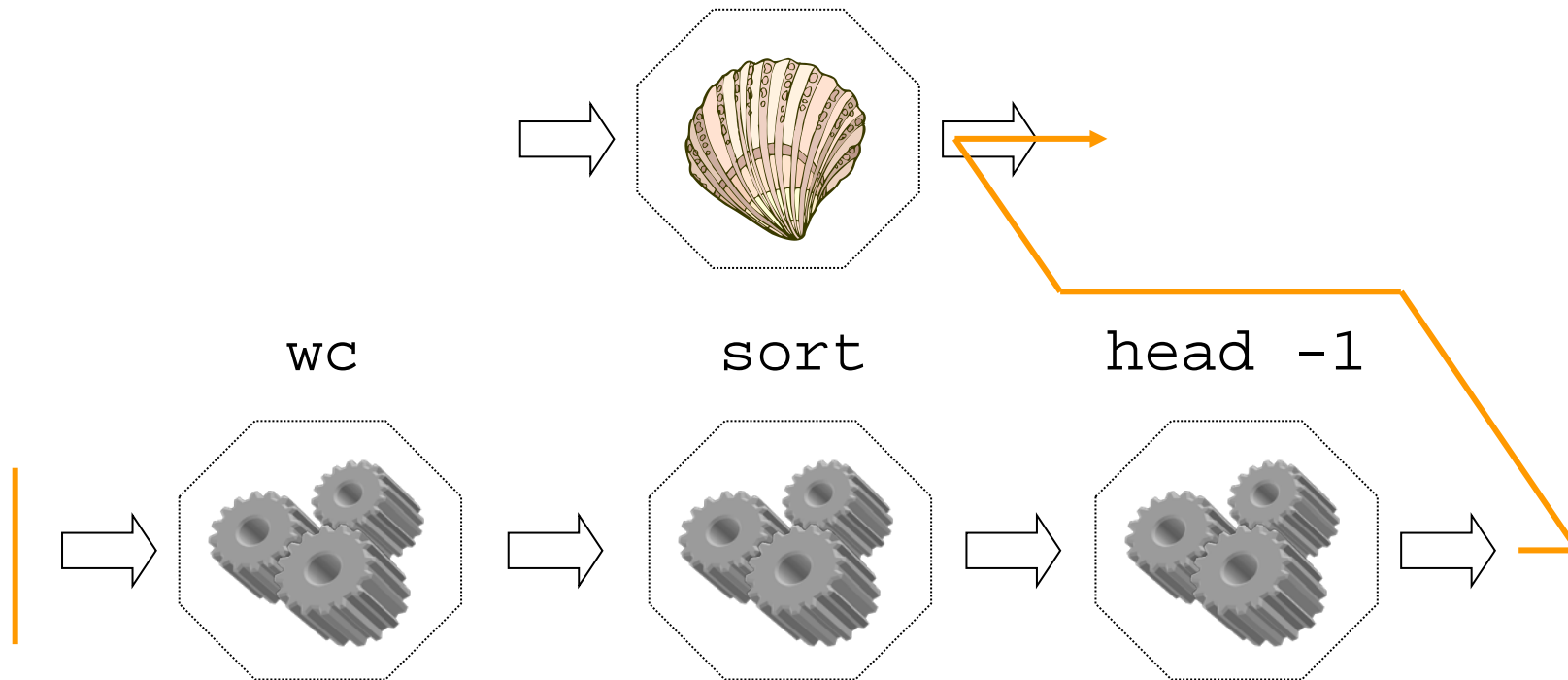
wc   sort   head -1

```
$ wc -l *.pdb | sort | head -1
```

This programming model called *pipes and filters*

This programming model called *pipes and filters*

A *filter* transforms a stream of input into a

stream of output

This programming model called *pipes and filters*

A *filter* transforms a stream of input into a

stream of output

A *pipe* connects two filters

This programming model called *pipes and filters*

A *filter* transforms a stream of input into a stream of output

A *pipe* connects two filters

Any program that reads lines of text from standard input, and writes lines of text to standard output, can work with every other

This programming model called *pipes and filters*

A *filter* transforms a stream of input into a stream of output

A *pipe* connects two filters

Any program that reads lines of text from standard input, and writes lines of text to standard output, can work with every other

You can (and should) write such programs

pwd	mkdir
cd	nano
ls	rm
.	rmdir
..	mv
	cp

pwd	mkdir	wc
cd	nano	sort
ls	rm	head
.	rmdir	
..	mv	
	cp	

pwd	mkdir	wc
cd	nano	sort
ls	rm	head
.	rmdir	*tail*
..	mv	*split*
	cp	*cut*
		uniq

pwd	mkdir	wc	*
cd	nano	sort	>
ls	rm	head	|
.	rmdir	*tail*	
..	mv	*split*	
	cp	*cut*	
		uniq	

pwd	mkdir	wc	*
cd	nano	sort	>
ls	rm	head	|
.	rmdir	*tail*	<
..	mv	*split*	*?*
	cp	*cut*	
		uniq	

created by

Greg Wilson

August 2010