OSU Oregon State UNIVERSITY

CENTER FOR GENOME RESEARCH & BIOCOMPUTING

“Introduction to Unix/Linux” INX_U18, Day 8, 2018-08-10

stdin, stdout, stderr, piping |, iterative filtering, grep, cat, UUOC

Learning Outcome(s):

Redirect the standard output to the standard input stream of another program.
Distinguish between the standard output and standard error streams
Utilize the tools grep and wc for basic analysis of bioinformatics data.

Matthew Peterson, OSU CGRB, matthew@cgrb.oregonstate.edu

Papilio zelicaon FASTA

- Common anise swallowtail butterfly of western North America
- `pz_cDNAs.fasta` from the book contains 471 *de-novo* assembled transcript sequences
  - ("*without a reference genome*" assembled DNA sequences, that are copied to RNA (mRNA); first step of gene expression)

fasta_stats

- A Python script from the book
- Generates statistics about a FASTA file it's supplied

```
./fasta_stats
Usage: fasta_dna_stats <fasta_file>
This script is for informational purposes only, and
requires that the input file be a DNA (As, Ts, Cs,
and Gs) FASTA-formatted file.
```

```
./fasta_stats pz_cDNAs_sample.fasta
```

# Only 2 sequences in this FASTA file

PZ7180000031590 example

1	2	3	4	5	6	7	8
PZ7180000031590	0.378	486	ACAAA	5	unit:ATTTA	10	pentanucleotide

2: GC content of 37.8% (vs. AT)

3: 486 base pairs (bp) long

4: Most common 5-bp sequence is ACAAA

5: This 5-bp sequence ACAAA appears 5 times

6: Longest perfect repeat sequence is ATTTA

7: and is 10bp long

8: caused by the pentanucleotide ATTTA appearing twice (**ATTTAATTTA**)

Oregon State UNIVERSITY

stdout redirection?

```
./fasta_stats pz_cDNAs_sample.fasta > pz_sample_stats.txt
Processing sequence ID PZ7180000031590
Processing sequence ID PZ7180000000004_TX
```

- Why was some output sent to the file (>) and some (2 lines) appeared on the terminal?
- Was everything from **stdout** redirected to the file?
- File contains everything but the above two lines?

```
less -S pz_sample_stats.txt
```

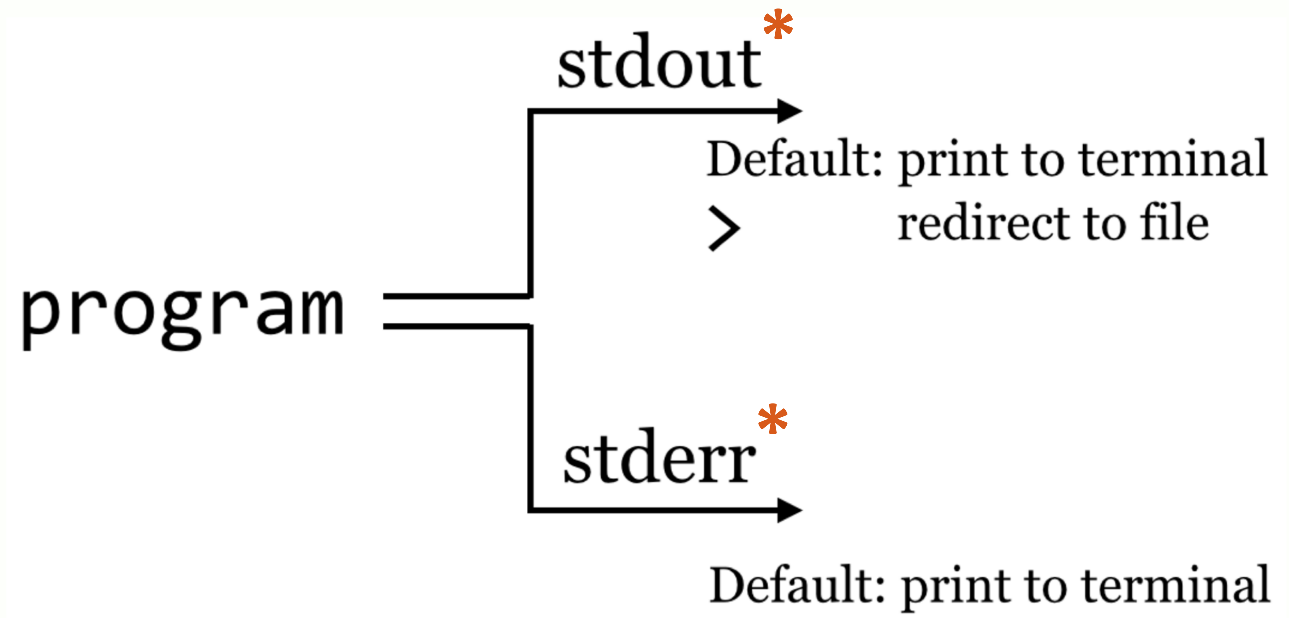
stderr Standard Error

- The information printed to the terminal is coming not from **stdout** but from a second stream, **stderr** ("standard error")
- By default **stderr** is printed to the terminal
- **stderr** usually contains warning messages or diagnostic information, e.g., the following was not part of the results, just an FYI:

```
Processing sequence ID PZ7180000031590
Processing sequence ID PZ7180000000004_TX
```

Oregon State UNIVERSITY

stdout and stderr



- Programs produce 2 streams:
  - **stdout** that can be redirected via **>**
  - **stderr** that by default is printed to the terminal

Capturing stderr to a file on tcsh

- As **2>** to capture **stderr** to a file only works on bash, there is a workaround for tcsh:

```
( ./fasta_stats pz_cDNAs_sample.fasta > pz_sample_stats.txt ) >& pz_sample_stats.err.txt
```

- Two independent redirects are run:
  - **(** `cmd > output` **)** results in the redirect of **stdout** first
  - The remainder of **stderr** is redirected by **>&**

Capturing stdout and stderr to a file

- If you wanted to capture both stdout and stderr to the same file you can use: **>&**

```
./fasta_stats pz_cDNAs_sample.fasta >& pz_sample_stats_ALL.txt
```

- Nothing appears on the terminal
- Everything is captured to the file

`grep` Extracting a pattern

```
grep '<pattern>' file
```

```
./fasta_stats pz_cDNAs.fasta > pz_stats.txt
grep 'unit:' pz_stats.txt
```

- This ignores the informational lines and only shows the output we care about (that contains the pattern '`unit:`' that we specified).
- To not match a pattern use **-v**

```
grep -v 'unit:' pz_stats.txt
```

grep results to a file

```
grep 'unit:' pz_stats.txt > pz_stats.table
less -S pz_stats.table
```

```
PZ832049        0.321   218     CTTAA   4       unit:CGT        6       trinucle
PZ21878_A       0.162   172     ATTAA   8       unit:ATT        6       trinucle
PZ439397        0.153   111     TTAAT   5       unit:GAAAT      10      pentanuc
PZ16108_A       0.157   191     ATTAA   7       unit:ATT        6       trinucle
PZ21537_A       0.158   82      TTATT   3       unit:ATT        6       trinucle
PZ535325        0.108   120     AATTA   6       unit:TA 6       dinucleotide
...
```

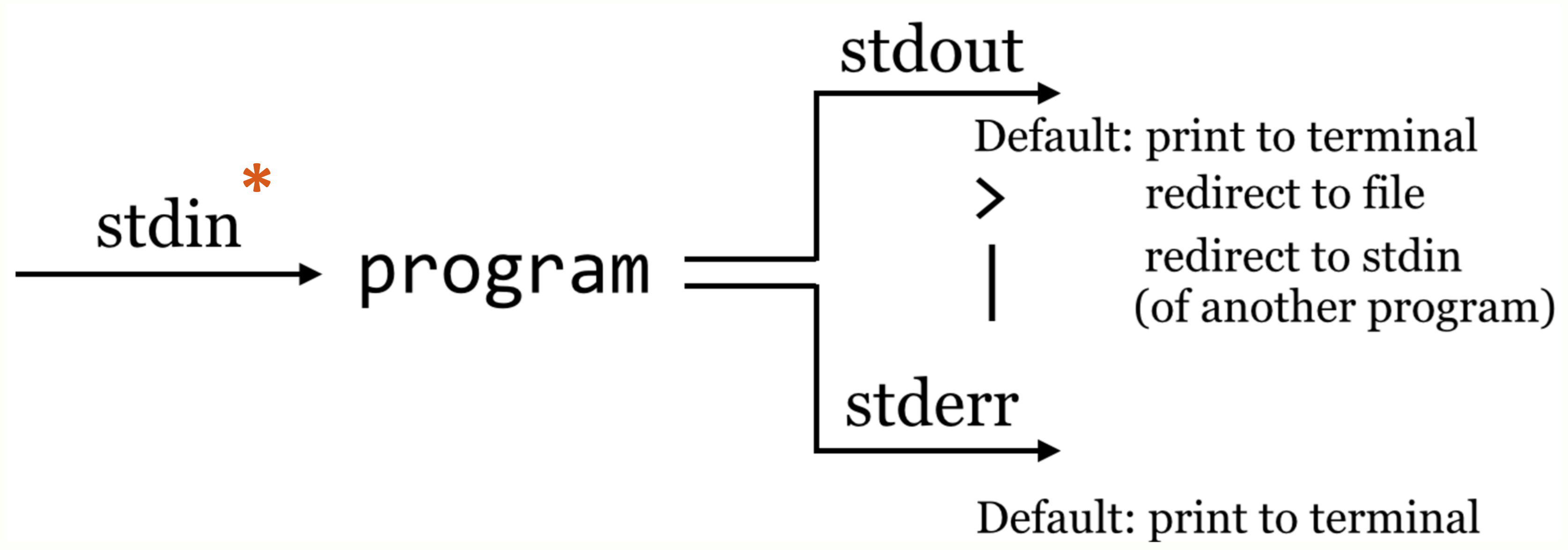
`wc` Word count (lines, words, chars)

```
wc <file>
wc pz_stats.table
```

To count only lines (not words or characters): `–l`

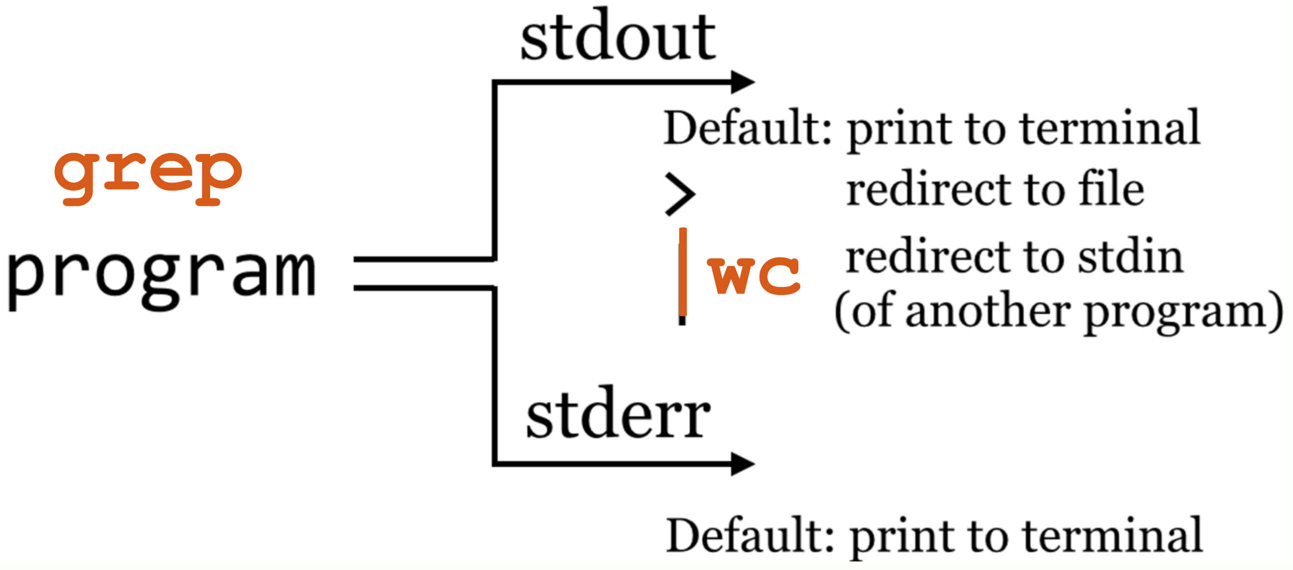
```
wc -l pz_stats.table
```

Reading from stdin (Standard input)


stdin*
program
stdout
Default: print to terminal
> redirect to file
| redirect to stdin (of another program)
stderr
Default: print to terminal

- **stdin** is a secondary ***input*** mechanism for programs (other than reading from files directly).
- By default, standard input (**stdin**) is not used.

stdin example using | ("pipe")

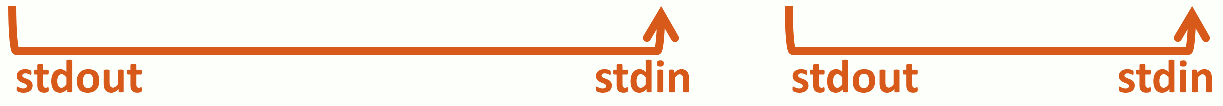

stdout
Default: print to terminal
> redirect to file
| wc redirect to stdin (of another program)
grep
program
stderr
Default: print to terminal

```
rm pz_stats.table
grep 'unit:' pz_stats.txt | wc
```

No new file was created! The **stdout** of **grep** was stored in a *temporary buffer* and sent to **stdin** of **wc**, which sent its results to **stdout**

More piping | examples

```
./fasta_stats pz_cDNAs.fasta | grep 'unit:'| wc
```



Two pipes (|) *chaining*:

- **fasta_stats**, which pipes (|) its std**out** to:
- std**in** of **grep**, which pipes (|) its std**out** to:
- std**in** of **wc**, which outputs to std**out** (terminal)

Note: std**err** is still printed to the terminal!

`cat` Concatenate files

```
cat <filename>
```

- It can be used to send files to **stdout**

```
cat <file1> <file2> <file3> …
```

- It can also be used to concatenate files to **stdout**

Q: Does the following work?

```
cat <file1> | grep 'pattern'
```

\ Commands over multiple lines

```
./fasta_stats pz_cDNAs.fasta \
| grep 'unit:' \
| wc
```

In tcsh appears as:

```
./fasta_stats pz_cDNAs.fasta \
? | grep 'unit:' \
? | wc
```

What happens if we up arrow ↑ post run?

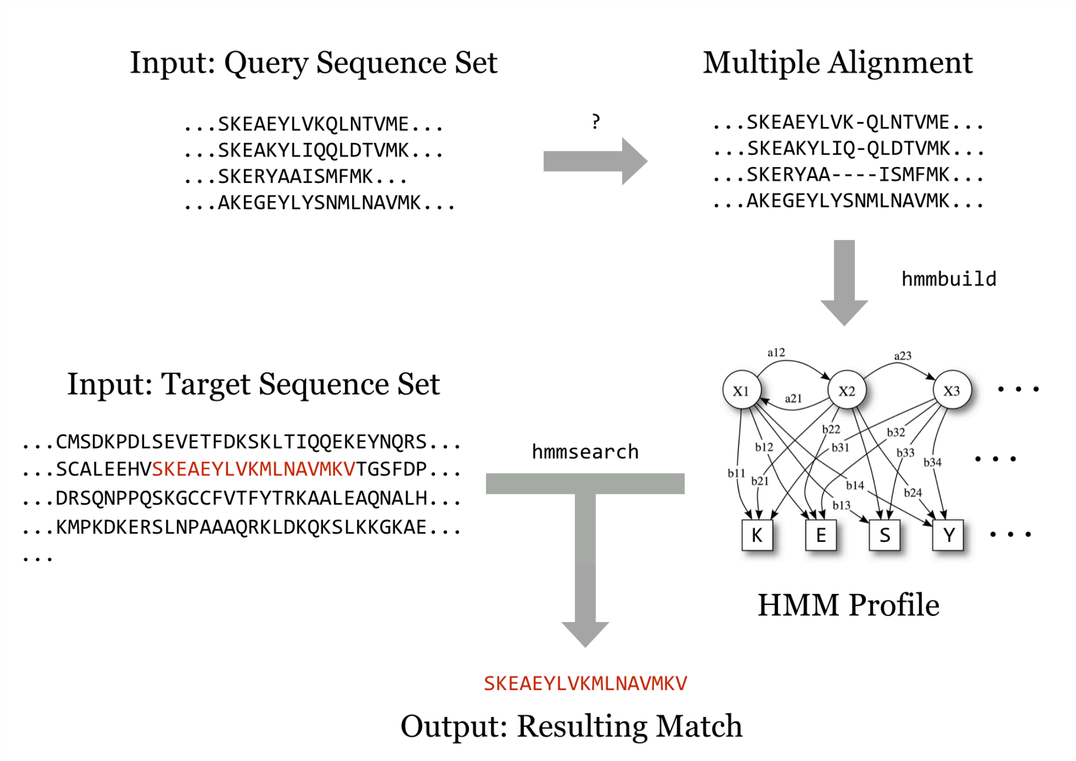
Pipelines!

```
cmd1 | cmd2 | cmd3 | cmd4 | cmd5
```

Chaining several commands together with **pipes** (**|**) on a **line** is is known as a… **pipeline**!

Oregon State UNIVERSITY

Pipelines in general



The term **pipeline** is often used for any series of steps from **input** data to **output** data, e.g., the Muscle/HMMER pipeline we built.

Oregon State University

Finding AT repeats

```
./fasta_stats pz_cDNAs.fasta \
| grep 'unit:AT' \
| less -S
```

```
PZ21878_A       0.162   172     ATTAA   8       unit:ATT        6       trinucle
PZ16108_A       0.157   191     ATTAA   7       unit:ATT        6       trinucle
PZ21537_A       0.158   82      TTATT   3       unit:ATT        6       trinucle
PZ7180000031590 0.378   486     ACAAA   5       unit:ATTTA      10      pentanuc
PZ7180000031597 0.287   403     ATTAT   6       unit:ATTTTG     12      hexanucl
PZ7180000025478 0.516   829     TGATG   18      unit:ATG        18      trinucle
...
```

- Matches **ATT** and **ATG**, not just **AT**
  - Not exactly what we wanted…

Filtering for AT dinucleotides

```
./fasta_stats pz_cDNAs.fasta \
| grep 'unit:AT' \
| grep 'dinucleotide'
| less -S
```

```
PZ7180000031598 0.209   81      AATAT   5       unit:AT 6       dinucleotide
PZ463243        0.226   97      TTGTA   3       unit:AT 4       dinucleotide
PZ7180000000106_T       0.246   1044    AAAAA   22      unit:AT 10      dinucleo
PZ17593_A       0.157   76      ATTAA   5       unit:AT 4       dinucleotide
PZ492422        0.144   90      ATTAA   5       unit:AT 4       dinucleotide
PZ22453_A       0.267   269     ATTAA   8       unit:AT 4       dinucleotide
...
```

- Good, now we're getting just the **ATs**

Counting AT dinucleotides via `wc -l`

```
./fasta_stats pz_cDNAs.fasta \
| grep 'unit:AT' \
| grep 'dinucleotide'
| wc -l
```

- "Processing sequence ID" to **stderr** is <u>not</u> counted
- Finally tally of **AT dinucleotides**: **22**

Q: What did we just do?

A: ***Iterative development!***

Oregon State UNIVERSITY

Script time! (reuse this code)

```
nano count_ATs.sh
```

```
#!/bin/tcsh

if ( $# != 1 ) then
    echo "Wrong number of parameters"
    echo "Usage: $0 <fasta_file>"
    exit
endif

setenv file $1
fasta_stats $file \
| grep 'unit:AT' \
| grep 'dinucleotide' \
| wc -l
```

$# Is the number of parameters (arguments) provided to the script

Oregon State UNIVERSITY

Command / Concept Review

- `>`
- `(cmd) >&`
- `|`
- `grep 'pattern'`
- `wc`
- `cat`
- `\`

`stdout`

`stderr`

`stdin`

Pipelines

Iterative development

Oregon State UNIVERSITY