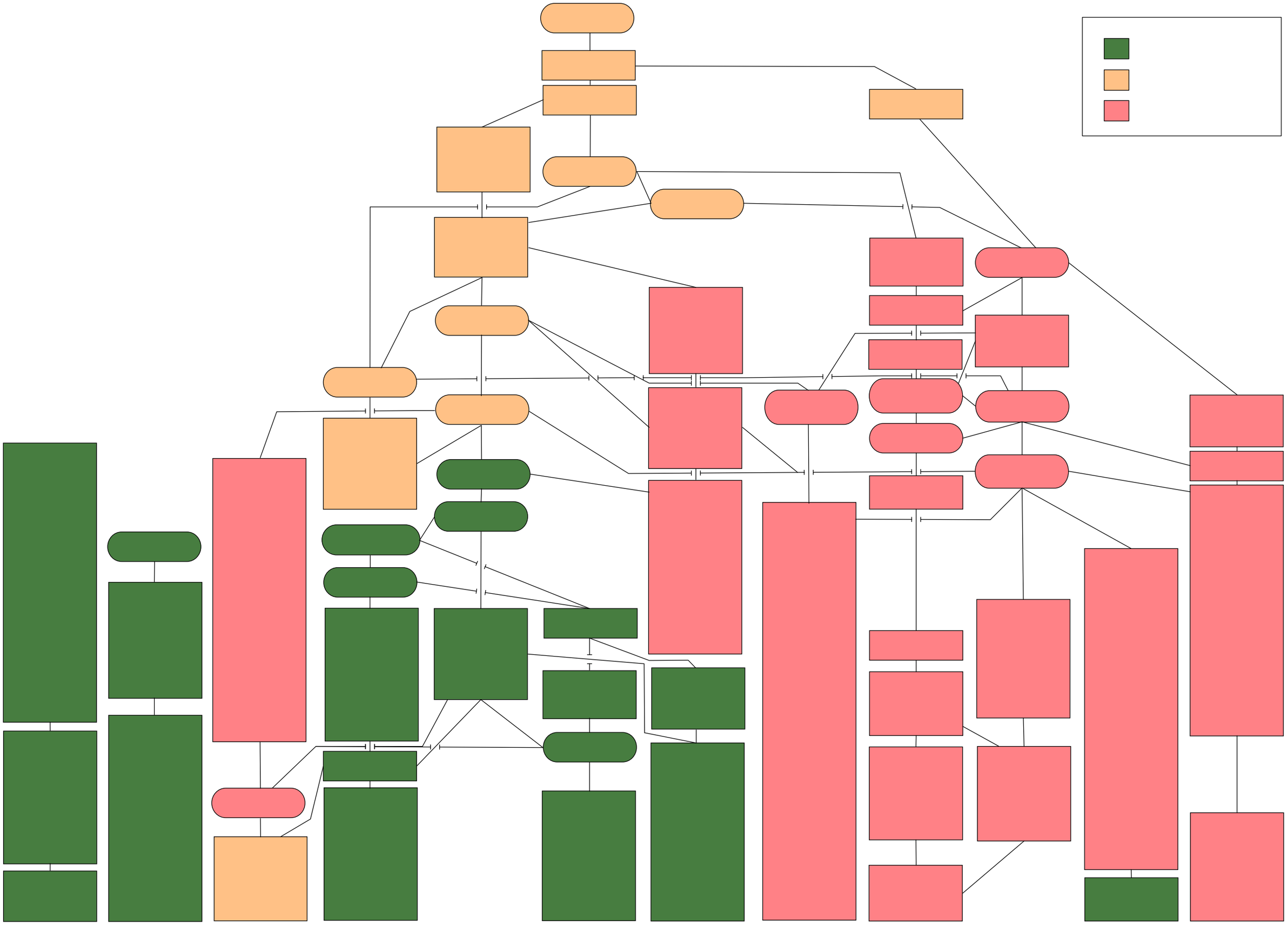
Unix History



What computers can run Unix?

What computers can run Unix?

- Apple OS X Macs
- Google's Android phones
- Most airplane entertainment systems
- Wireless internet routers

The Terminal Window

The Terminal Window

Make it comfortable to work in:

- Resize the window
- Change your font size
- Open multiple terminal windows

Obtain a cheat sheet

google “unix commands”

In UNIX everything is a file organized in a hierarchy

Paths

/home/catchen/working

Paths, cont

This shell view of the nested directories shell, research, seq, and radtags…..

…. is equivalent to this GUI view of the same directories

And the `radtags` directory is uniquely identified by its path:

`/home/ubuntu/shell/research/seq/radtags`

Create a series of directories

Create a series of directories

```
% mkdir shell        % cd shell          % ls
```

Absolute and relative paths

How do I get to the Hotel Zlaty Andel?

Absolute and relative paths

How do I get to the Hotel Zlaty Andel?

Absolute and relative paths

How do I get to the Hotel Zlaty Andel?

Paths

/home/catchen/working

Relative Path

/home/catchen/research

Relative Path

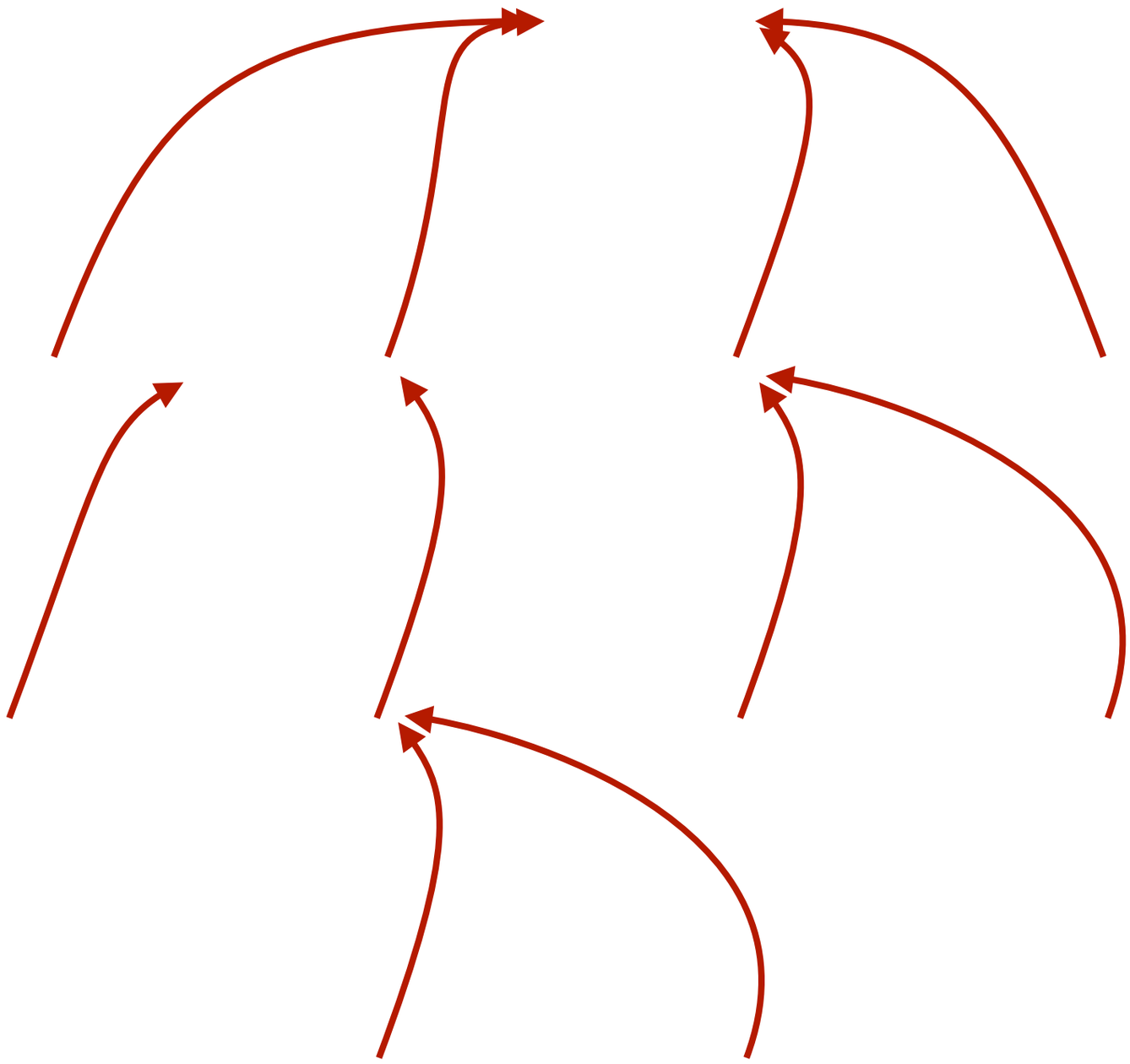
```
../working
```

Special files -- '*dot*'

Special files -- '*dot*'

Special files -- '*dot dot*'

Special files -- '*dot dot*'



Absolute and relative paths

Special Files

dot

dot dot

Absolute and relative paths

Special Files

dot

dot dot

```
% ls .
```

```
% ls ..
```

```
% ls ../../
```

Binary programs - ls, cp, mkdir, etc.

Binary programs - ls, cp, mkdir, etc.

```
% ls /bin
```

Relative and absolute paths

A shortcut to your ‘home’, tilde:

~

Moving through the filesystem:

cd

Knowing where you are:

pwd

```
% ls ~/          % cd ~/          % cd          % pwd
```

Relative and absolute paths

`/home/ubuntu/shell/research/seq/radtags`

Create a series of directories under shell:

Relative and absolute paths

`/home/ubuntu/shell/research/seq/radtags`

Create a series of directories under shell:

`% ls .`

`% ls ..`

`% ls ../../`

Relative and absolute paths

`/home/ubuntu/shell/research/seq/radtags`

Create a series of directories under shell:

```
% ls .
% cd ~/
```

```
% ls ..
% cd shell/research
```

```
% ls ../../
% pwd
```

Are you typing? You're doing it wrong.

Tab-completion:

- Tab once to complete uniquely
- Tab twice to see all possible completions

Up-arrow:

- Previous commands can be found by pressing "up-arrow"

'history'

Are you typing? You're doing it wrong.

Tab-completion:

- Tab once to complete uniquely
- Tab twice to see all possible completions

Up-arrow:

- Previous commands can be found by pressing "up-arrow"

'history'

```
% ls c <tab>
```

```
% ls c <tab><tab>
```

Are you typing? You're doing it wrong.

Tab-completion:

- Tab once to complete uniquely
- Tab twice to see all possible completions

Up-arrow:

- Previous commands can be found by pressing "up-arrow"

'history'

```
% cd /etc
% ls c <tab>
```

```
% pwd
% ls c <tab><tab>
```

Three variants to `ls`

`ls -l`	`ls -la`	`ls -lh`
provides a *long* listing	includes *all* files, even hidden files	displays file sizes in *human* readable numbers

Four ways to view a text file

`more`	`head`	`tail`	`cat`
view a text file one screen full at a time	view the top 15 lines of a file	view the last 15 lines of a file	spit the whole file at once
space-bar: scroll q: quit	-n num controls the number of lines	-n num controls the number of lines	

Explore the file hierarchy

1. Move to the directory /etc
    - What is the first line of the files '`hosts`' in the directory `/etc`?
    - What is the relative file path to get to `/var/log` from here?
    - What is the absolute path?
2. Move to the directory `/var/log/`
    - What is the contents on line 73 of the dmesg file?
    - Without changing directories, what is the second line of the `cpuinfo` file in the `proc` directory?
        - What is the command to read this file with a relative path?
        - An absolute path?
3. Move back to the root, what directories do you see?
4. Move back home, what are three ways to move home from the root?

Download example files using `wget`

Return to the directory in your home called ‘shell’.

TSV file:
`http://creskolab.uoregon.edu/cesky/batch_1.genotypes_1.loc.gz`

FASTQ file:
`http://creskolab.uoregon.edu/cesky/s_1_sequence.txt.gz`

Tar Archive:
`http://creskolab.uoregon.edu/cesky/samples.tar.gz`

What is a tar archive?

tar = tape archive

Compress / Decompress

`gzip` / `gunzip`

batch_1.genotypes_1.loc.gz

s_1_sequence.txt.gz

Gzipped Tar archive

`tar xvfz`

samples.tar.gz

Tar archive

`tar xvf`

samples.tar

The FASTQ File Format

FASTA

```
>HWI-ST0747:162:C03AJACXX:3:1108:19763:106771 1:N:0:
TTTGTCTGCAGGGGGACACGTCAAAGTCAAACGCAGGCAAGTTTGTGTTTATGTCCAGTGGATCTTTTGATTTT
ACATACTGCAGGGTCAGGAGGATTATCTCCTCTGCAAGGTAACGCCTGCTGTAACCGTTGTTCTTCATCCTTTT
CCTAACTGCAGGGCTGTCTTGTCAGGTCTGACAAGACATATGCAGGGCTCAATTTGAGATAATTGCTCAATATA
```

FASTQ

```
@HWI-ST0747:162:C03AJACXX:3:1108:19763:106771 1:N:0:
TTTGTCTGCAGGGGGACACGTCAAAGTCAAACGCAGGCAAGTTTGTGTTTATGTCCAGTGGATCTTTTGATTTT
+
<?@DDDDDHFHHFBB@GGIACFHGGHBGHGCDHBEAHACHI=@CH.=7ACAHHADECDBCC66(6>@C>5@CACCA
```

The FASTQ File Format, ctd

```
@HWI-ST0747:162:C03AJACXX:3:1108:19763:106771 1:N:0:
TTTGTCTGCAGGGGGACACGTCAAAGTCAAACGCAGGCAAGTTTGTGTTTATGTCCAGTGGATCTTTTGATTTT
+
<?@DDDDDHFHHFBB@GGIACFHGGHBGHGCDHBEAHACHI=@CH.=7ACAHHADECDBCC66(6>@C>5@CACCA
```

Quality Scores

```
  SSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSS.....................................
  ........................XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX.....................
  ............................IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII.....................
  ..............................JJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJ.....................
 LLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLL.....................................
  !"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvwxyz{|}~
  |                          |    |        |                                 |                     |
 33                         59   64       73                               104                   126

 S - Sanger        Phred+33,  raw reads typically (0, 40)
 X - Solexa        Solexa+64, raw reads typically (-5, 40)
 I - Illumina 1.3+ Phred+64,  raw reads typically (0, 40)
 J - Illumina 1.5+ Phred+64,  raw reads typically (3, 40)
    with 0=unused, 1=unused, 2=Read Segment Quality Control Indicator (bold)
    (Note: See discussion above).
 L - Illumina 1.8+ Phred+33,  raw reads typically (0, 41)
```

ASCII values 64 - 104 = 0 - 40

http://en.wikipedia.org/wiki/FASTQ_format

The FASTQ File Format, ctd

```
@HWI-ST0747:162:C03AJACXX:3:1108:19763:106771 1:N:0:
TTTGTCTGCAGGGGGACACGTCAAAGTCAAACGCAGGCAAGTTTGTGTTTATGTCCAGTGGATCTTTTGATTTT
+
<?@DDDDDHFHHFBB@GGIACFHGGHBGHGCDHBEAHACHI=@CH.=7ACAHHADECDBCC66(6>@C>5@CACCA
```

Quality Scores

```
  SSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSS.....................................
  ........................XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX.....................
  ............................IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII.....................
  ..............................JJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJ.....................
 LLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLL.....................................
  !"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvwxyz{|}~
  |                         |    |        |                                     |                     |
 33                        59   64       73                                   104                   126

 S - Sanger        Phred+33,  raw reads typically (0, 40)
 X - Solexa        Solexa+64, raw reads typically (-5, 40)
 I - Illumina 1.3+ Phred+64,  raw reads typically (0, 40)
 J - Illumina 1.5+ Phred+64,  raw reads typically (3, 40)
    with 0=unused, 1=unused, 2=Read Segment Quality Control Indicator (bold)
    (Note: See discussion above).
 L - Illumina 1.8+ Phred+33,  raw reads typically (0, 41)
```

ASCII values 64 - 104 = 0 - 40

‘F’ = 70

http://en.wikipedia.org/wiki/FASTQ_format

The FASTQ File Format, ctd

```
@HWI-ST0747:162:C03AJACXX:3:1108:19763:106771 1:N:0:
TTTGTCTGCAGGGGGACACGTCAAAGTCAAACGCAGGCAAGTTTGTGTTTATGTCCAGTGGATCTTTTGATTTT
+
<?@DDDDDHFHHFBB@GGIACFHGGHBGHGCDHBEAHACHI=@CH.=7ACAHHADECDBCC66(6>@C>5@CACCA
```

Quality Scores

```
  SSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSS.....................................
  ........................XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX.....................
  ............................IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII.....................
  ..............................JJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJ.....................
 LLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLL.....................................
  !"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvwxyz{|}~
  |                         |    |        |                                     |                     |
33                        59   64       73                                   104                   126

S - Sanger        Phred+33,  raw reads typically (0, 40)
X - Solexa        Solexa+64, raw reads typically (-5, 40)
I - Illumina 1.3+ Phred+64,  raw reads typically (0, 40)
J - Illumina 1.5+ Phred+64,  raw reads typically (3, 40)
   with 0=unused, 1=unused, 2=Read Segment Quality Control Indicator (bold)
   (Note: See discussion above).
L - Illumina 1.8+ Phred+33,  raw reads typically (0, 41)
```

ASCII values 64 - 104 = 0 - 40

‘F’ = 70 70 - 64 = **14**

http://en.wikipedia.org/wiki/FASTQ_format

The FASTQ File Format, ctd

```
@HWI-ST0747:162:C03AJACXX:3:1108:19763:106771 1:N:0:
TTTGTCTGCAGGGGGACACGTCAAAGTCAAACGCAGGCAAGTTTGTGTTTATGTCCAGTGGATCTTTTGATTTT
+
<?@DDDDDHFHHFBB@GGIACFHGGHBGHGCDHBEAHACHI=@CH.=7ACAHHADECDBCC66(6>@C>5@CACCA
```

Quality Score	Probability of incorrect base call	Base call accuracy
10	1 in 10	90%
20	1 in 100	99%
30	1 in 1000	99.9%
40	1 in 10000	99.99%

http://en.wikipedia.org/wiki/Phred_quality_score

The FASTQ File Format, ctd

```
@HWI-ST0747:162:C03AJACXX:3:1108:19763:106771 1:N:0:
TTTGTCTGCAGGGGGACACGTCAAAGTCAAACGCAGGCAAGTTTGTGTTTATGTCCAGTGGATCTTTTGATTTT
+
<?@DDDDDHFHHFBB@GGIACFHGGHBGHGCDHBEAHACHI=@CH.=7ACAHHADECDBCC66(6>@C>5@CACCA
```

70 - 64 = 14

Quality Score	Probability of incorrect base call	Base call accuracy
10	1 in 10	90%
20	1 in 100	99%
30	1 in 1000	99.9%
40	1 in 10000	99.99%

http://en.wikipedia.org/wiki/Phred_quality_score

Count raw reads:

```
wc -l s_1_sequence.txt

grep "@" s_1_sequence.txt
grep -c "@" s_1_sequence.txt

grep -v "@" s_1_sequence.txt
grep -v -c "@" s_1_sequence.txt
```

Count reads with barcode:

```
grep -c "^CGATA" s_1_sequence.txt
```

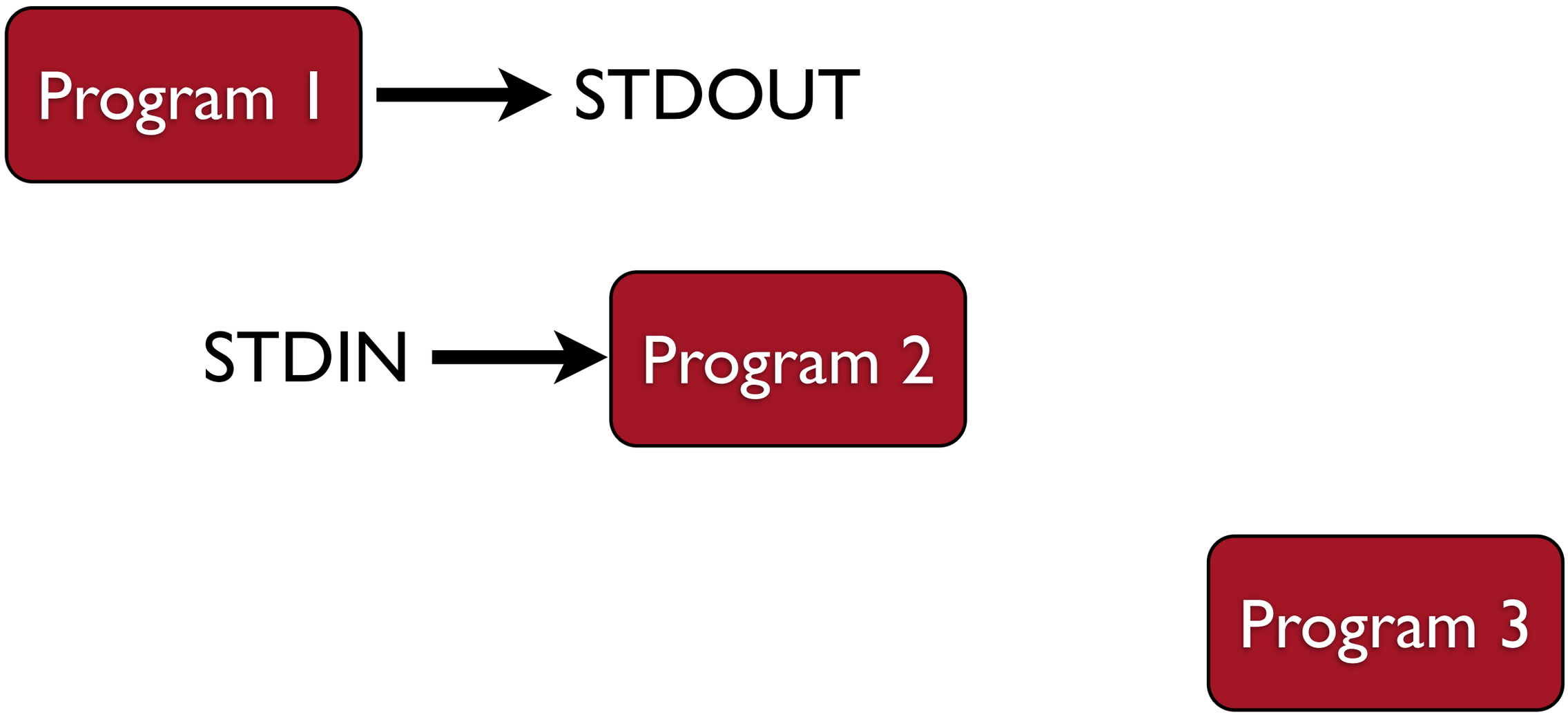
Special Files

STDIN, STDOUT, STDERR

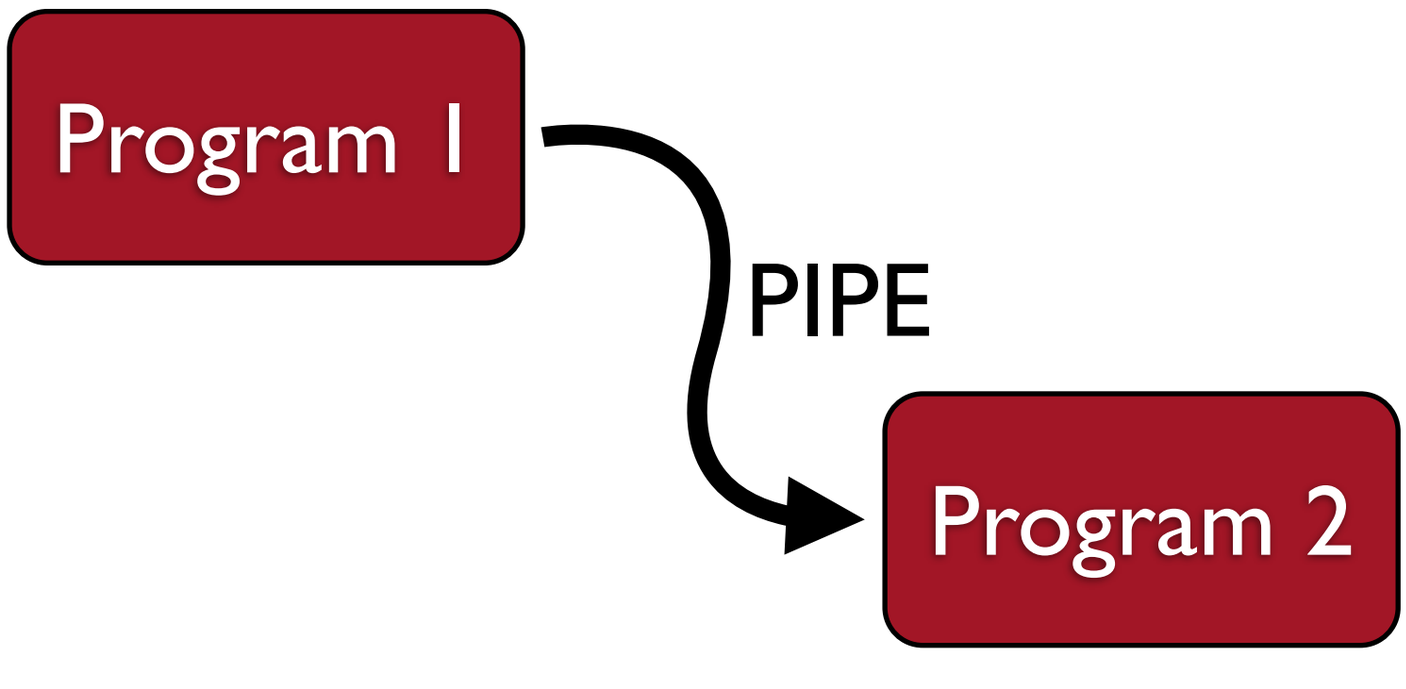
The Shell's Killer App: **Pipes**

The Shell's Killer App: **Pipes**, ctd.

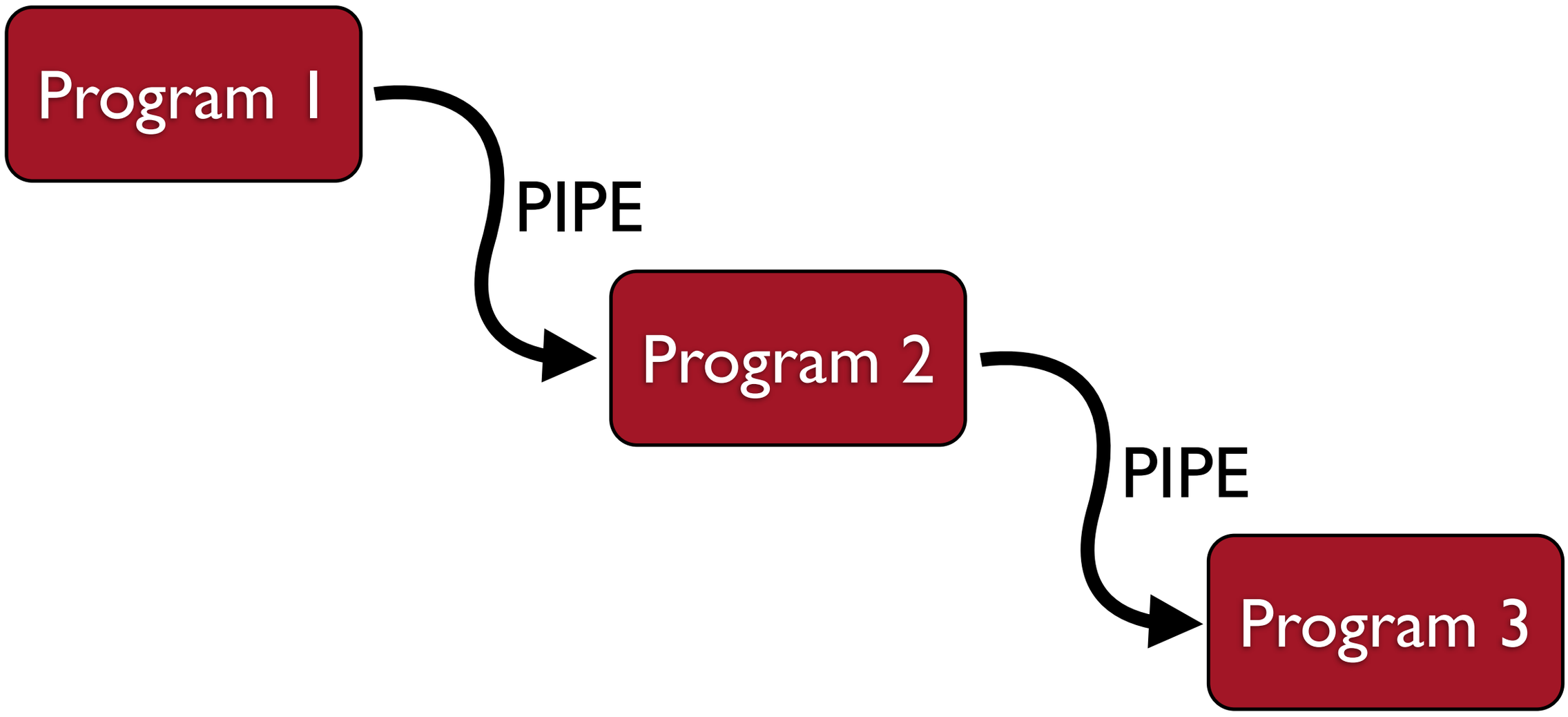
The Shell's Killer App: **Pipes**, ctd.



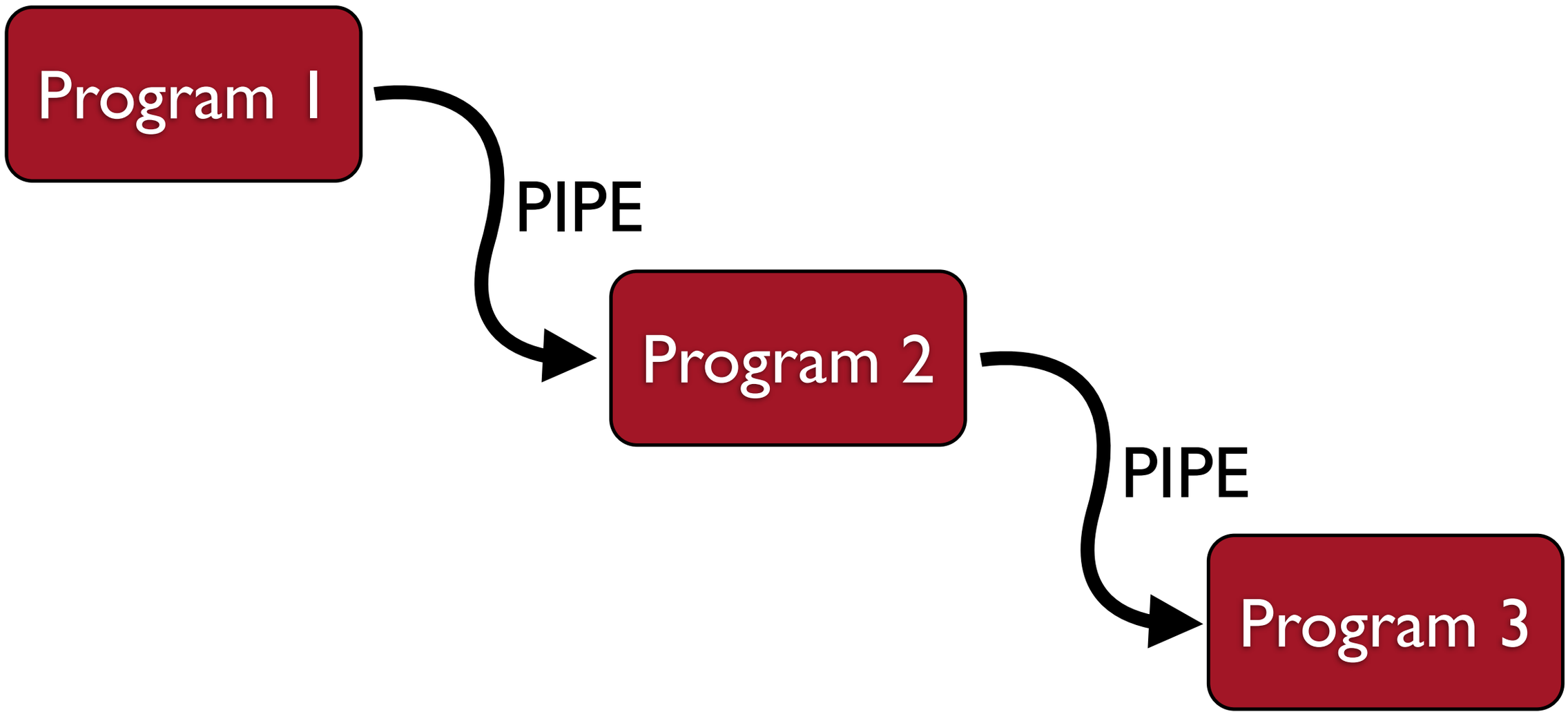
The Shell's Killer App: **Pipes**, ctd.



The Shell's Killer App: **Pipes**, ctd.



The Shell's Killer App: **Pipes**, ctd.



So what is the purpose of the program **`cat`**?

cut

```
cut -f 10 batch_1.genotypes_1.loc
```

cut, capture the output

```
cut -f 1-10 batch_1.genotypes_1.loc > genos
```

cut, pipe the output to grep

```
cut -f 2 batch_1.genotypes_1.loc | grep -c "nnxnp"
```

```
cut -f 1-10,15,17 batch_1.genotypes_1.loc | grep "nnxnp" > genos2
```

Examine a marker, translating the output

```
cat batch_1.genotypes_1.loc | tr "     " "," | grep "^96053"
```

s_1_sequence.txt.gz

1. Decompress the file
2. Count the number of raw reads (250,000)
3. Count the number of reads with barcode `CGATA` (19,501)
4. Capture all FASTQ records for `ACCAT` into a file called sample_01.fq (you should get 18352 records, 73408 lines)
5. Determine the count of all barcodes in the file

```
286 CTAGT
7900 TCAGA
10659 ACTGC
10931 TGACC
11536 GAGAT
11871 CTGAA
14409 CGGCG
14508 TGGTT
18226 GAAGC
18352 ACCAT
18375 TCGAG
19501 CGATA
23012 AATTT
26336 GCATT
31136 CTAGG
```

`ls`
`gunzip`
`man`
`more`
`cat`
`wc`
`head`
`cut`
`grep`
`sort`
`uniq`
`>`
`|`

1. Use **`head`** when building a command, **`cat`** once the command is working
2. Look at the **-n** option for the **head** command, the **-l** option for **wc**
3. The "^" character means "must occur at beginning of line" in a grep search
4. Look at the **grep** options: **-c**, **-v, -A**, **-B**
5. Read the man pages for **sort** and **uniq** to learn how to combine them

cut

```
cut -f 10 batch_1.genotypes_1.loc
```

cut, capture the output

```
cut -f 1-10 batch_1.genotypes_1.loc > genos
```

cut, pipe the output to grep

```
cut -f 2 batch_1.genotypes_1.loc | grep -c "nnxnp"
```

```
cut -f 1-10,15,17 batch_1.genotypes_1.loc | grep "nnxnp" > genos2
```

Examine a marker, translating the output

```
cat batch_1.genotypes_1.loc | tr "    " "," | grep "^96053"
```

Count raw reads:

```
wc -l s_1_sequence.txt

grep "@" s_1_sequence.txt
grep -c "@" s_1_sequence.txt

grep -v "@" s_1_sequence.txt
grep -v -c "@" s_1_sequence.txt
```

Count reads with barcode:

```
grep -c "^CGATA" s_1_sequence.txt
```