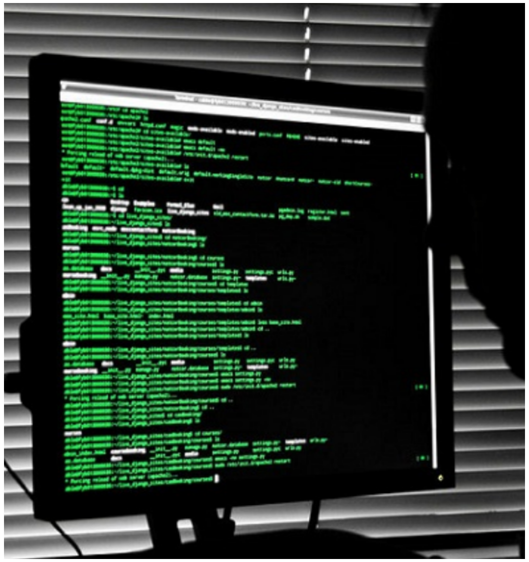
Day 3 Introduction to UNIX command-line



Serghei Mangul, Ph.D.
University of California, Los Angeles

@serghei_mangul

Summary of Day 2

- file permissions
- cat
- wc
- >, >>, <
- pipeline
- ln –s
- grep
- regex
- sed

awk

- awk is both a
  - programming language
  - text processor

whitespace (spaces, tabs, etc.) to separate fields

- parsing and manipulating **tabular** data
  - iterates through the entire file line-by-line

```
awk '{action_to_take}' <file_to_parse>
```

awk : Simple Uses

action_to_take

```
awk '{print}’ <file_to_parse>
```

```
awk '{print $1}’ <file_to_parse>
```

column number

```
awk '{print $1”\t”$3}’ <file_to_parse>
```

delimiter

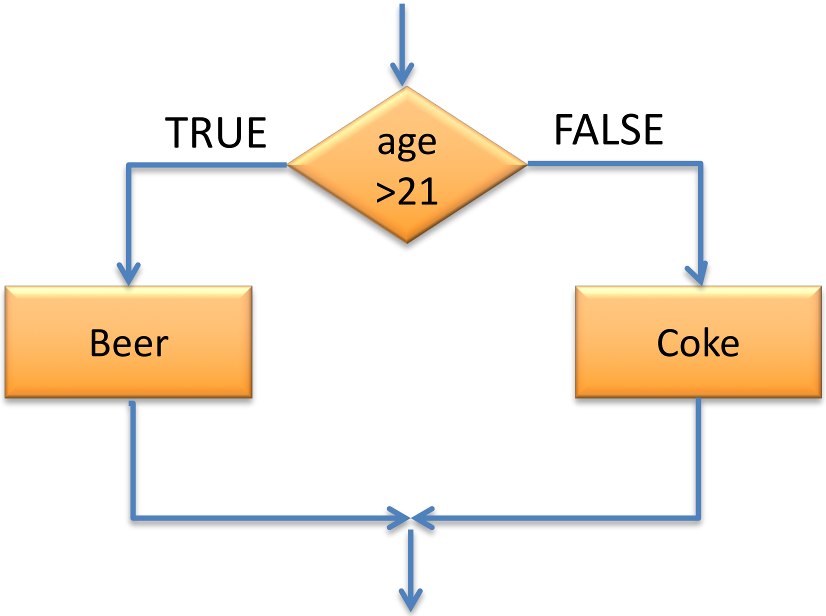
Let’s practice

```
awk '{print}' hg19.gtf
awk '{print $1}’ hg19.gtf
awk '{print $4}' hg19.gtf
awk '{print $1"\t"$4} hg19.gtf
awk '{print $1"\t"$4*$5}' hg19.gtf
```

awk : If statement

condition

```
if(age>21) then
    print("Beer Please")
```



awk : If statement

Print those employees who actually worked

```
Beth         4.00      0
Dan          3.75      0
```

tabular data

```
awk '{if($3>0) print}' emp.data
```

Data type

- Numbers
- **Text**

```
1         clothing         3141
[redacted]
1         textbooks        21312
2         clothing         3252
[redacted]
2         supplies         2242
2         textbooks        15462
```

Print information about computers only

```
awk '{if($2=="computers") print}' sales.dat
```

use double quote

Text

Let’s practice!

```
awk '{if($3>0) print}' emp.data
awk '{if($3>0) print $1"\t"$2*$3}' emp.data
```

```
Beth     4.00    0
Dan      3.75    0
Kathy   4.00    10
Mark    5.00    20
Mary    5.50    22
Susie    4.25    18
```

Let’s practice!

```
awk '{if($4>50) print}’ hg19.gtf
awk '{if($1=="chr2") print}’ hg19.gtf
awk '{if($1=="chr2") print $1"\t"$3}’ hg19.gtf
```

```
chr2	hg18_knownGene_GnfAtlas2	exon	237538	237602	0.000000	-	.	gene_id "204019_s_at"; transcript_id "uc002qvu.1";
chr2	hg18_knownGene_GnfAtlas2	exon	239731	239852	0.000000	-	.	gene_id "204019_s_at"; transcript_id "uc002qvu.1";
```

awk : sum

- Calculate sum of a particular column

```
awk '{sum+=$2} END { print sum}' emp.data
```

New variable

“+=” means to add the value to the variable sum

Do it until the end of the file

A variable is like a **box** where we can store a value and reuse this same value multiple times in our program.

Let’s practice!

```
awk '{sum+=$4} END {print sum}’ hg19.gtf

awk ' END {print NR}' hg19.gtf
wc –l hg19.gtf

awk '{sum+=$4} END {print sum/3000}’ hg19.gtf
awk '{sum+=$4} END {print sum/NR}’ hg19.gtf
awk '{if($1=="chr2") sum+=$4} END {print sum}’ hg19.gtf
```

```
chr2   hg18_knownGene_GnfAtlas2   exon   237538  237602  0.000000   -  .   gene_id "204019_s_at"; transcript_id "uc002qvu.1";
chr2   hg18_knownGene_GnfAtlas2   exon   239731  239852  0.000000   -  .   gene_id "204019_s_at"; transcript_id "uc002qvu.1";
```

Sort

- will rearrange the lines in a text file so that they are sorted, numerically and alphabetically.

```
sort [OPTION]... [FILE]...
```

Options :

- -n - numerical ordering
- -k - sort by a particular column

Sort a file

regex.txt

```
beat
brat
boat
bat
banana
```

```
sort regex.txt
```

Sort a text file

```
sort -n regex.txt
```

Sort a file with numbers

Sort by column

```
sort -k 2 sales.dat
```

Column number

```
sort -k 3n sales.dat
```

sales.dat

```
1     clothing     3141
1     computers     9161
1     textbooks     21312
2   clothing     3252
3   …
```

Uniq

- Removes duplicate lines from a file*

```
uniq [OPTION]... [INPUT]
```

Options :

- -c - how many times each line occurred
- -d - print only duplicated lines

*assumes that the file is sorted

Let’s practice

```
$ sort -n numbers.txt >numbers_sort.txt
$ uniq numbers_sort.txt
$ sort numbers.txt | uniq
$ uniq -d numbers_sort.txt
$ uniq -c numbers_sort.txt
```

2 7

a line of the file

How many times it occurs

numbers.txt

3
4
5
7
2
1
6
7

Shell scripts

- If you have a bunch of commands you’d like to automate, you can put them on separate lines of a file.

My first shell script!

Following steps are required to write shell script:

Use any **vi** editor like to write shell script.

After writing shell script set execute permission for your script

My first shell script!

```
vi script.sh
```

```
#!/bin/bash
echo "My first script”
```

SH

How to run the script :

```
chmod u+x script.sh
./script.sh
```

More scripting

- Let’s create a bash script which will split <gtf> into files corresponding to every chr (2,3,21), save every file in separate directory called chr${i}_gtf.

More scripting

vi script.sh

```
#!/bin/bash
echo "My first script"

mkdir chr2_gtf
mkdir chr3_gtf
mkdir chr21_gtf

grep "chr2\s" $1 >chr2_gtf/chr2.gtf
grep "chr3\s" $1 >chr3_gtf/chr3.gtf
grep "chr21\s" $1 >chr21_gtf/chr21.gtf
```

How to run the script :

```
./script.sh hg19.gtf
```

Do biomedical researchers have to become programmers?

+

Google

stackoverflow

SEQanswers
the next generation sequencing community

=

*provided in the class

*free and easy to use

Good luck!