8/22/24

UNIX

- 1970s at AT&T
- Thomas, Ritchie, McIlory, Ossanna

- Kernel - interacts with hardware, memory management, task scheduling, and file management
- Shell - interprets command and calls the program
- Commands
  - Cp, mv, cd, cat, grep
- Files and directories

- Root partition that leads to rest of file-system management

Command line - explainshell

- ls : lists whats in current directory
- -t : sorts by time
- -h : marks legible for humans
- -o :
- -r : reversed sorting
- Ctrl c or d : exits
- ~ : home directory shortcut
- ls / : all folders
- cd : change directory
- pwd : present working directory to check where you are
- mkdir : creates directory
- touch : creates empty file
- echo : prints text to location
- more : shows what’s in the file
- cat : prints contents of file
- cd .. : goes up
- cd home : goes to home directory
- cd ssh : initial directory

8/27/24

- **Repeat a one-line command:** for i in {1..6}; do echo "Hello World"; done >>file.txt
- curl : download directly from website onto termina;
- grep -c “>” **or** grep --count “>” **or** cat example.fasta | grep ">" | wc -l **:** counts occurrences of > (relation to lines)
- grep -o “T” example.fasta | wc -l

Grep created in 11/1973 by Ken Thompson

G: global search
Re: regular expression
P: print

grep [option] pattern file
-e : extra functions
-i : ignore upper/lower case
-w : exact word
^ : start of a line
$: end of a line
[ ] : any of the enclosed characters

- scp kite_data.txt ssh@SF-Laptop : to upload file to terminal

8/29/24

- Grep command to print all lines with “AT”
- Man grep
- Grep -n “x” file : prints lines that contain x
- grep -o "AT" example.fasta | wc -l
- grep -c "^kite" kite_data.txt : counts number of lines that starts with kite NOT Kite
- grep -o "[Kk]ite|red" kite_data.txt | wc -l
- sed : stream editor, transforms text into simple programming language
  - Replaces specified with new thing in file
  - - -help
  - s = substitute, g = global
  - Only first occurrence if g removed
- echo "Hello World" | sed 's/World/There/g'
- sed 's/[a-z]/\U&/g' kite_data.txt
- sed -i '/Three/d' file.txt: deletes lines
- Sed ‘/^$/d’ empty_lines.txt : deletes empty lines

Lab

- egrep : -o shows matches
- Grep -cv “x” file : shows how many lines that don’t contain x

9/3/24

- To delete empty lines: grep -v or -e ‘^$’ file.txt
- To delete all “white spaces”: grep -v or -e ‘[[:space:]]*$’ file.txt

Paths - organizational systems

- / - root (lowest path to access)
- Echo $PATH - finds which directories your shell is set to check for executable files
- Top - shows what’s running and how much mem and processors its using
- Export PATH=$PATH:/new/path

TSV - tab separated
CSV - , separated

- Sed ‘s/,/\t/g’ file.csv >file.tsv
- Sed ‘s/\t/,/g’ file.tsv >file.csv
- cat file.tsv | tr -s ‘\t’ ‘,’ >file.csv

cut [options] file.txt

- -f selects a field, either 1-12 or 1,12 depending on what you’re deleting
- -c characters

sort [options] file.txt

- -t set the delimiter
- -k 2 sort column
- -k 2,3 sort multiple columns
- -n sort numerically
- -r reverse order sort
- -u drops duplicates
- -b ignore leading blanks
- -d dictionary order
- -f ignore case
- No options = sort alphabetically from furthest left character

uniq [options] file.txt : only works on sorted files and adjacent lines

- -c count lines for unique value
-
    - sort -k1 doppelganger_names.txt | uniq -c | wc -l (unique lines)
    - sort -k1 doppelganger_names.txt | uniq -d | wc -l (duplicated lines)
    - cut -f2 doppelganger_names.txt | sort | uniq -c

9/5/24

Wget [options] URL (upload from the web) (also curl)

- -V = version
- -h = help
- -b = background
- -o

zip example.zip names_game.txt
To view: vim example.zip   exit: shift q
Unzip example.zip : to extract
Unzip ‘*.zip’
Gunzip *.gz
Binzip2 *.bz2

#extract

```
tar -xzf tar-file-name.tar.gz
tar -xjf tar-file-name.tar.bz2
tar -zxvf data.tar.gz or .tgz (lists files -v)
tar -xjvf data.tar.bz2 (lists files -v)
tar -xvpzf somefilename.
```

```
#compress
tar -jcvf tar.bz2 file
tar -zcvf .tar.gz file
tar -csfv = ‘compress zip file verbose’
```

-c = compress   -x = extract

tr [option] SET1 [SET2]
 -c : complements the set of characters in a string

```
tr “'[a-z]' '[A-Z]' file : lower to uppercase
cat name_game.csv | tr -d '[:space:]' : removes white space
cat name_game.csv | tr -d ‘[Dd]’ : will delete all d’s in a file
```

Printf [-v var] format [arguments]

- - - help
- --version
- %d : signed decimal numbers
- \n : new line

nano : Ctrl x to exit

<u>Lab</u>

<u>9/12/24</u>

Bash

- Bash -x file.sh variable
- Always start with #!/bin/bash
- fi or end to end/finish
- Constants - values that don’t change

nucleotides: A, T, C, G, U

.fna - ATCG (fasta nucleotide file - contigs/reads)
.ffa - fasta feature nucleotide (nucleotide for protein ORFs)
.fastq - quality encoded

Fruits=('apple' 'banana' 'orange') : makes array
$fruits
echo ${Fruits[1]} : for element tied to 1 to print
echo ${Fruits[*]} : For all elements listed
echo ${#Fruits[@]} : counts the number of elements
echo ${Fruits[-1]} : prints the last element
Fruits+=('watermelon') : adds element to array
Fruits=(“${Fruits[@]}” “watermelon”) : adds element to array
Fruits=(${Fruits[@]/ap*/}) : removes apple as an element
Fruits=("${Fruits[@]}" "${Veggies[@]}") # Concatenate

9/17/24
Loops and Conditionals

grep "ATG" example2.fasta --color : prints all sequences, colors the section grep is searching for
grep ‘ATG.*ATG’ examples.fasta >extracted.fasta : prints line with 2 instances of ATG
grep ‘\(.*ATG\)\{2\}’ example2.fasta >extracted fasta : prints line with 2 instances of ATG
grep ‘\(.*ATG\)\{3\}’ test.fna | more
Variables - stores data, integers

Conditional loops: **for i in file.*; do command $i done**

```
#!/bin/bash
for i in 1 2 3 4 5
do
 echo "welcome $i times"
done
```

Or

```
#!/bin/bash
for i in {1..5}
do
 echo "welcome $i times"
done
```

In C:

```
for ((c=1 ; c<5 ; c++)
do
 echo “welcome $c times”
Done
```

9/19/24
Gsub is the same as s in awk
Head - top 5
Tail - bottom 5

grep -o "Steven.Jose" file.tsv | wc -l : counts and prints the number of times that Steven is left of Jose

grep ‘\(.*ATG\)\{3\}’ test.fna | more : prints line that has more than 1 ATG

Sam: Sequence alignment map file between the reads and reference

Print specific line numbers
Cat -n doppleganger_names.txt | grep “david” –color
sed -n ‘1p’ doppelganger_names.txt : grabs line 1
sed -n ‘1p:2p’ doppelganger_names.txt : grabs line 1 and 2
sed -n '3,7p' doppelganger_names.txt : grabs lines 3-7

Cat –number doppelganger_names.txt | head …….

```
#!/bin/bash

num1=$1
num2=$2

If [ $num1 != $num2]; then
echo “less than or equal”
else
echo “not less than or equal”
fi
```

```
#!/bin/bash

echo "enter first number"
read num1

echo "enter second number"
read num2

if [ $num1 -ge $num2 ]; then
 echo "greater than or equal"
else
 echo "Not greater than or equal"
```

```
#!/bin/bash

home=’pwd’
echo =$home

for i in *.fasta;
do
    Grep ”.”  “$i” | wc -l
    Head “$i”
done
```

—---

```
array=(‘apple’ ’banana’ ’cherry’)

for item in “${array [@]}”;
do
   echo $item
done                                                    also know in C-style
```

—---

Iterate over the indices of the array

```
arr=(“apple” “banana” “cherry”)

for index in “${!arr[@]}”;
do
echo “$index -> ${!arr[
```

—---

```
arr={[2]=”apple” [4]=”banana” [9]=”cherry”)
2 -> apple
4 -> banana
9 -> cherry
```

—--

```
#!/bin/bash

x=1
while [ $x -le 5 ]
do
  echo "Welcome $x time"
x=$(( $x + 1 ))
Done
```

—--

While loops will run until the condition becomes false
Until loops will run until the condition becomes true

9/26/24
GitHub

10/1/24
Specific file:

```
#!/bin/bash
input=$1
Sed ‘s/,/\t/g’ $input
Function print_to_terminal(){
        Echo “ file has converted” >$(tty)
        Echo -n “wise control!” >$(tty)
}

output=$(print_to_terminal)
```

All files

```
#!/bin/bash

for i in *csv;                                      #i is the placeholder
do
Sed ‘s/,/\t/g’ “$i” >$(basename “$i” .csv).tsv      #”$i” is calling the file into the placeholder
done
Function print_to_terminal(){
        Echo “ file has converted” >$(tty)
        Echo -n “wise control!” >$(tty)
}

output=$(print_to_terminal)
```

```
#!/bin/bash
#variables
file=$1
range1=$2
range2=$3

#range finder
sed -n $2,$3"p" $file
 #run it by doing: bash *filename* *script name* *range*
```

Saving a corrupted file

```
#!/bin/bash
diff $1 $2 --color
```

Find sequences with >4 N’s in the sequence + Print the header of the sequence

```
#!/bin/bash

input1=$1

grep -B1 ‘\(.*N\)\{4\}’ $1                    #-B1 is below header
```

Diff way:

```
#!/bin/bash

input1=$1
N_dectector(){
        Grep -B 1 “NNNN” $input
}
output=$(N_dectector)
Echo $output
```

Ways to convert tsv to csv
Translate

Slurm_sbatch manages scripts on the cluster
Slurm is a workload/package manager

3

10/3/24

Review FASTA info/file types
Printf
Echo -n etc
Sed : brackets for all special characters
White space vs empty lines
Perl with grep, awk, and sed
cut
Paths
Convert tsv to csv in one lines and scripts
Perl -pi
Prints output
For loops - bash and c style, python?

While loops
slurm/sbatch