Week 12

tar

- c: create
- v: verbose
- f: use the following file
- z: compress
- x: extract
- t: list contents
- r: append

common usage:

> `tar cvfz new-name.tar.gz file-to-tar` # compress and archive the file
>
> `tar xvfz tar-file.tar.gz` # extract compressed archive file

grep

d.f. globally search a regular expression and print

Examples:

1. `grep -in ".* interface" regular-expressions.txt` # `-i` ignore case and `n` print line numbers
2. `grep -wn express regular-expressoins.txt` # `-w` select only words
3. `grep -c express regular-expressions.txt` # `-c` counts the number of occurences
4. `grep -o express regular-expressions.txt` # `-o` print only the occurences

Color for grep `export GREP_OPTIONS='--color=auto'`

Find every file with the name perl

```
for i in *; do
  file $i | grep perl
done
```

Find all Executable files

`ls | grep .exe`

Find unique words

```
for i in `cat awt.txt`; do
  echo $i
done | sort -u > $HOME/sortedWords.txt
...
# OR Sort by unique words
done | sort | uniq -c > $HOME/sortedWords2.txt
...
# OR Sort by Frequency
done | sort | uniq -c | sort > $HOME/sortedWords3.txt
```

AWK

Pattern scanning and processing language

Useful with awk

`\~` compare with

`!~` doesn't compare with

Usage

`gawk 'program' filenames`

i.e.

Print out field one for /etc/passwd output

> `cat /etc/passwd | gawk -F: '{print $1}'`

`NR` - Number of Records

> `gawk '{print NR, $0}' datafile01`

Awk program

```
{
  # awk01
  nc += length ($0)
  nw += NF # Number of fields
}
END {print NR, nw, nc}
```

Run

> `gawk -f awk01 awmt.txt`

AWK Practice

```
awk '/core/ {print $0}' datafile01 # print the line with core in it
...
ls -al | awk '$5 > 200 {print $8}' # print the 8th field with the 5th field having
...
awk '$1 ~ /camelot/ {$print $0}' datafile01 # compare field 1 with camelot and then
...
awk '$3 !~ /1200/ {print $1}' datafile01
...
awk '{print NR, $0}' datafile01 # Print new records with entire line
...
awk '{print NF}' datafile01 # prints the number of fields for each line
...
awk '/eagle/ {system("ls -al")}' datafile01 # if it sees eagle, run ls -al
...
awk '/eagle/ {system("notepad")}' datafile01 # if it sees eagle, run notepad
```

AWK and CSV's

```
# First replace commas with spaces
tr ',' ' ' < Elements.csv > elementsv2 # transform command to replace commas with s

# Remove quotes
tr -d '\"' < elementsv2 > elementsv3 # delete quotes

# check number of fields of each line
awk '{print NF}' elementsv3

# Print the line if it does not have 10 fields
awk '(NF != 10) {print $0}' elementsv3
```

Heredocs

```
#! /bin/bash
  DATE=`date`
  cat << EOF
  date: $DATE

  Now is the time
  for all good programmers
  EOF
```