Lecture-5:
Awk- A Pattern Scanning and Processing Language

Tongping Liu

Tongping.Liu@utsa.edu

Assignment 1

Some questions about homework:

- Part 1: only files (and its directory) are listed
- Part 2: how to check line by line? Change to an array at first
- Part 3:

Awk Introduction

Awk is a utility for processing structured data

- Anything that has multiple entries in the same fields
- Example:
  - 'Contacts' file where each contact has a name & phone number
  - Awk could be used to print just the phone numbers

Splits a file into fields (columns) and operates on each row (line)

- Useful for processing log files, experimental data, etc

More information:

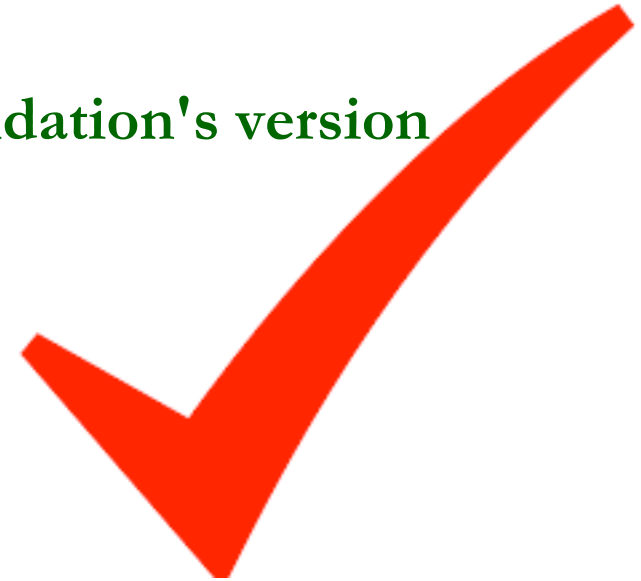
http://www.gnu.org/software/gawk/manual/gawk.html

Different Types of AWK

AWK - the (very old) original from AT&T

NAWK - A newer, improved version from AT&T

GAWK - The Free Software foundation's version



awk - processing columns of data

awk [-F fs] [-v var=value] ['prog' | -f progfile] [file ...]

field separator | variables | script in ‘ ‘ or in file | data file

awk -F “,” '{print $2}' data.txt

awk -F “,” -f script.awk data.txt

If the field separator is SPACE or TAB, there is no need to specify.

No input file

Awk 'program':

- Awk applies the program to the standard input
- Continues only you hit "CTRL+d".

- DEMO: awk '{ print }'

Different Ways to Run a command

1. Running this command in the command line
   - awk "BEGIN { print \"Don't Panic!\" }”
   - Awk ‘BEIN{ print “Don’t Panic!”}’

2. Run use a file
   - Awk –f 1noinput.awk

3. Run as an awk script
   - #!/usr/bin/awk –f
   - chmod +x 1noinput.awk
   - ./1noinput.awk

Awk examples

Value of a column = $1, $2, $3 for column 1, 2, 3, etc

To print out the second column:

```
awk   '{print $2}'   data.txt
```

Output:

10
20
30
40
50

data.txt

```
1 10 xyz 100
2 20 abc 200
3 30 def 300
4 40 ade 400
5 50 f2d 500
```

Awk example – Using the file

Can also put awk script into a file

awk -f simple.awk data.txt

simple.awk

```
{print $2}
```

Can have multiple sections of code

- **Separate with braces {}**
- **BEGIN, END for start, end of file**

sections.awk

```
BEGIN {print "Column 4"}
{print $4}
END {print "The end."}
```

```
Column 4
100
200
300
400
500
The end.
```

Awk examples

Value of a column = $1, $2, $3 for column 1, 2, 3, etc

To print out the second and fourth column:

```
awk   '{print $2 "," $4}'   data.txt
```

single quotes around program

double quotes for text

data.txt

```
1 10 xyz 100
2 20 abc 200
3 30 def 300
4 40 ade 400
5 50 f2d 500
```

Output:

10, 100
20, 200
30, 300
40, 400
50, 500

Variables

User variables

- string, numeric

Program variables

Variable	Meaning
$0	Current record
$1-$n	Fields in the current record
FILENAME	Current input file name (null for standard input)
FS	Input field separator (default: SPACE or TAB)
NF	Number of fields in the current record
NR	Record number of the current record
OFS	Output field separator (default: SPACE)
ORS	Output record separator (default: NEWLINE)
RS	Input record separator (default: NEWLINE)

awk exercise

data.txt

```
1 10 xyz 100
2 20 abc 200
3 30 def 300
4 40 ade 400
5 50 f2d 500
```

Write the awk command to print the word “Sum:”, followed by the sum of columns 1 and 4

```
awk 'BEGIN{print "Sum:"}{print $1+$4}' data.txt
```

Examples

```
awk '{print $3, $1}' cars

awk '/chevy/ {print $3, $1}' cars
```

“back slash” indicates the pattern to search for

```
$ cat cars
plym    fury     1970
chevy   malibu   1999
ford    mustang  1965
volvo   s80      1998    102     9850
ford    thundbd  2003    15      10500
chevy   malibu   2000    50      3500
bmw     325i     1985    115     450
honda   accord   2001    30      6000
ford    taurus   2004    10      17000
toyota  rav4     2002    180     750
chevy   impala   1985    85      1550
ford    explor   2003    25      9500
```

```
plym    1999 chevy
chevy   2000 chevy
ford    1985 chevy
1998 volvo
...
```

More Examples

Print the longest length of a file

```
awk '{ if (length($0) > max) max = length($0) }
    END { print max }' data
```

```
expand data | awk '{ if (x < length()) x = length() } END
{ print "maxim
```

Change the Tab to spaces at first

Print every line that is longer than 80 characters

```
awk 'length($0) > 10' data
```

More Examples

Print those lines with more than 1 field

```
awk '{ NF > 1 } ' data
```

Print the total number of bytes used by files

```
ls -l files | awk '{ x += $5 } END { print "total bytes: "
x }'
```

Count the lines in a file data

```
awk 'END { print NR }' data
```

Print the even-numbered lines in the data file

```
awk 'NR % 2 == 0' data
```