Chapter 9 - AWK

Philipp Moritzer - 21170004

---

awk

- An output formatting language
- Performs operations primarly on exisitng text
- Pattern-Matching program
- Takes 2 inputs
    - Command
    - Data

```
$ awk '{print $0}' /etc/passwd
```

- Output is sent to standard output

Extracting data

```
$ awk -F":" '{print "username: " $1 "\t\t\t user id: " $3}' /etc/passwd
```

Output:

```
$ awk -F":" '{print "username: " $1 "\t\t\t user id: " $3}' /etc/passwd
username: root                  user id: 0
username: daemon                        user id: 1
username: bin                   user id: 2
username: sys                   user id: 3
username: adm                   user id: 4
username: lp                    user id: 71
username: uucp                  user id: 5
username: nuucp                 user id: 9
username: smmsp                 user id: 25
username: listen                        user id: 37
username: gdm                   user id: 50
username: webservd                      user id: 80
username: postgres                      user id: 90
username: svctag                        user id: 95
username: unknown                       user id: 96
username: nobody                        user id: 60001
username: noaccess                      user id: 60002
username: nobody4                       user id: 65534
username: test                  user id: 100
```

```
-F # switch : field seprator, default is blank space
- $1 # first field
- $3 # third field
- $0 # whole line
```

Command 2 parts

- Patterns
  - Patterns are matched with the line of the data file
  - No pattern provided: Any line is always executed
- Actions
  - Have several constructions
  - Each instructions include =, if, while, for, print, printf

Patterns

- Consists of
  - String of text
    - Example: /unix/
  - Regular Expression
    - Example: [a-z]
- Escape sequence at Printf
  - \n (new line), ' (single quotation mark), \\ backslash, \0 null, \b backspace

Standard Variables

- FS: field separator
- NR: current line
- NF: number of words on a specific line
- FILENAME: name of input file
- RS: separator for each line in a file

Programming with awk

```
$ awk -f print.awk /etc/passwd
```

- Awk file:

```
- BEGIN {FS=":"; total=0}
- {total+=1; print $1 $3}
- END {print total NR}
```

- Awk needs three sections:
  - BEGIN: executed only at the beginning
  - Main: executres once for each and every line
  - BEGIN and END can be omitted

Example.awk

```
BEGIN { num = 0; a[2] =0 ; a[3] = 0 }
{ sum=0
 for ( i=2 ; i<=NF ; i++ ) {
 sum += $i; a[i] += $i
}
 printf("%s : sum %d avg is %.2f \n", $1, sum, sum/(NF-1))
num++
}
END {
for ( i=2 ; i<=4 ; i++ ) {
 printf "%d th sum %d average %.2f \n", i, a[i], a[i]/NR
 }
}
```

Output:

```
$ awk -f print2.awk data.txt
simon : sum 260 avg is 86.67
nho : sum 250 avg is 83.33
inho : sum 250 avg is 83.33
mio : sum 250 avg is 83.33
minh : sum 250 avg is 83.33
2 th sum 457 average 91.40
3 th sum 355 average 71.00
4 th sum 448 average 89.60
```