Shell
awk

What is awk ??

- The word awk is derived from the names of its inventors awk is actually

Aho

Weinberger

Kernighan

What is awk ?? (cntd…)

- “awk is a programming language designed to make many common information retrieval and text manipulation tasks easy to state and to perform.”
- Simply put, awk is a programming language designed to search for, match patterns, and perform actions on files.

How it works??

- Syntax:

  awk '/search pattern1/ {Actions}

  /search pattern2/ {Actions}' file
- search pattern is a regular expression.
- Actions – statement(s) to be performed.
- file – Input file.
- Single quotes around program is to avoid shell not to interpret any of its special characters.

awk Working Methodology

- awk reads the input files one line at a time.
- For each line, it matches with given pattern in the given order, if matches performs the corresponding action.
- If no pattern matches, no action will be performed.
- In the above syntax, either search pattern or action are optional, But not both.

awk Working Methodology (cntd…)

- If the search pattern is not given, then awk performs the given actions for each line of the input.
- If the action is not given, print all that lines that matches with the given patterns which is the default action.

awk Working Methodology (cntd…)

- Empty braces with out any action does nothing. It wont perform default printing operation.
- Each statement in Actions should be delimited by semicolon.

AWK examples

```
$cat employee.txt
100  Thomas     Manager    Sales          $5,000
200  Jason      Developer  Technology     $5,500
300  Sanjay     Sysadmin   Technology     $7,000
400  Nisha      Manager    Marketing      $9,500
500  Randy      DBA        Technology     $6,000
```

AWK examples (cntd…)

- Default behavior

```
$ awk '{print;}' employee.txt
100  Thomas  Manager    Sales       $5,000
200  Jason   Developer  Technology  $5,500
300  Sanjay  Sysadmin   Technology  $7,000
400  Nisha   Manager    Marketing   $9,500
500  Randy   DBA        Technology  $6,000
```

AWK examples (cntd…)

- **Print the lines which matches with the pattern.**

```
$ awk '/Thomas/
> /Nisha/' employee.txt
100  Thomas  Manager    Sales       $5,000
400  Nisha   Manager    Marketing   $9,500
```

AWK examples (cntd…)

- **Print only specific fields.**

$ awk '{print $2,$5;}' employee.txt

Thomas $5,000

Jason $5,500

Sanjay $7,000

Nisha $9,500

Randy $6,000

>>>> $0??

AWK examples (cntd…)

- **Initialization & Finalization with BEGIN & END.**

```
$ awk 'BEGIN {print "Name\tDesignation\
   tDepartment\tSalary";}
> {print $2,"\t",$3,"\t",$4,"\t",$NF;}
> END{print "Report Generated\n----------";
> }' employee.txt
```

AWK examples (cntd…)

- **Initialization & Finalization with BEGIN & END.**

Name Designation Department Salary

Thomas Manager Sales $5,000

Jason Developer Technology $5,500

Sanjay Sysadmin Technology $7,000

Nisha Manager Marketing $9,500

Randy DBA Technology $6,000

Report Generated

AWK examples (cntd…)

- **Find the employees who has employee id greater than 200.**

```
$ awk '$1 >200' employee.txt
300  Sanjay  Sysadmin   Technology  $7,000
400  Nisha   Manager    Marketing   $9,500
500  Randy  DBA         Technology   $6,000
```

AWK examples (cntd…)

- **Print the list of employees in Technology department.**

```
$ awk '$4 ~/Technology/' employee.txt
200  Jason   Developer  Technology  $5,500
300  Sanjay  Sysadmin   Technology  $7,000
500  Randy   DBA        Technology  $6,000
```

>>can you try $4==“Technology”

AWK examples (cntd…)

- **Print number of employees in Technology department.**

```
$ awk 'BEGIN { count=0;}
$4 ~ /Technology/ { count++; }
END { print "Number of employees in Technology
    Dept =",count;}' employee.txt
Number of employees in Tehcnology Dept = 3
```

awk Built-in Variables

NF : Number of fields in current line/record

NR : Ordinal number of current line/record

FS : Field Separator (Also -F can be used)

OFS : Output Field Separator (default=blank)

ORS: Output Record Separator

FILENAME : Name of current input file

awk some more examples

```
$ cat test1.txt
    Continent:Val
    AS:12000
    AF:9800
    AS:12300
    NA:3400
    OC:12000
    AF:500
    AS:1000
```

awk some more examples (cntd…)

- Print the first field, excluding the first line

```
$ awk -F ":" 'NR!=1 {print $1}' test1.txt
    AS
    AF
    AS
    NA
    OC
    AF
    AS
```

awk some more examples (cntd…)

- **Print first and second field, except the first row**

```
$ awk -F ":" 'NR!=1 {print $1,$2}' test1.txt
AS 12000
AF 9800
AS 12300
NA 3400
OC 12000
AF 500
AS 1000
```

awk some more examples (cntd…)

- **Print first and second field, except the first row, and setting the OFS as | (pipe)**

```
$ awk -F ":" 'BEGIN{OFS="|"} NR!=1 {print $1,$2}' test1.txt
AS|12000
AF|9800
AS|12300
NA|3400
OC|12000
AF|500
AS|1000
```

awk some more examples (cntd…)

- **Print first and second field, except the first row, and setting the OFS as | (pipe)**

```
$ awk 'BEGIN{FS=":"; OFS="|"} NR!=1 {print $1,$2}' test1.txt
AS|12000
AF|9800
AS|12300
NA|3400
OC|12000
AF|500
AS|1000
```

Awk some more examples (cntd...)

- **'Or' and 'AND' together**

```
$ awk -F ":" '($1=="AS" || $1=="OC") && $NF >
11000 {print}' test1.txt
AS:12000
AS:12300
OC:12000
```

awk some more examples (cntd…)

- **Reading from STDOUT**

```
$ cat test1.txt | awk -F ":" '!/Continent/ {print $1}' | sort | uniq
AF
AS
NA
OC
```

awk some more examples (cntd…)

- **Sum of 2nd fields**

```
 awk -F ":" 'NR!=1 {sum+=$NF} {print sum}' test1.txt
12000
21800
34100
37500
49500
50000
51000
```

awk some more examples (cntd…)

- **Sum of 2nd fields**

  $ awk -F ":" 'NR!=1 {sum+=$NF} END {print sum}' test1.txt
  51000

awk some more examples (cntd…)

- Average of 2nd field, as first field is excluded, (NR-1) instead of NR for total number of items

```
$ awk -F ":" 'NR!=1 {sum+=$NF} END {print sum/(NR-1)}'
    test1.txt
    7285.71
```

awk some more examples (cntd…)

- Set 2nd value as 0 where first field is "AS"

```
awk -F ":" 'BEGIN {OFS=":"} $1=="AS" {$2=0} {print}' test1.txt
```

```
Continent:Val
AS:0
AF:9800
AS:0
NA:3400
OC:12000
AF:500
AS:0
```

awk some more examples (cntd…)

- If statements…
- eg3.txt =
  The cow jumped over the moon
- if statements
  - `awk '{if ($1 == "he") { print $0; }}' eg3.txt`
    (empty)
  - `awk '{if ($1 ~ "he") { print $0; } else { … }}' eg3.txt`
    The cow jumped over the moon

awk some more examples (cntd…)

- for loops

```
awk '{for (j=1; j <= NF; j++)
      { print $j }
    }' eg3.txt
```

awk some more examples (cntd…)

- for loops

```
awk '{for (j=1; j <= NF; j++)
      { print $j }
    }' eg3.txt
```

Can you do??

- **Print the sum of fields in every line.**
- **Print the sum of fields in all lines.**
- **Replace every field by its absolute value.**

- **Count the total number of fields (words) in a file.**
- **Find the line containing the largest (numeric) first field.**

Can you do?? (cntd…)

- **Print the last field of the last line.**
- **Print every line with more than 4 fields.**
- **Print every line where the value of the last field is greater than 4.**

```
awk '{ s = 0; for (i = 1; i <= NF; i++) s = s+$i; print s }’

awk '{ for (i = 1; i <= NF; i++) s = s+$i }; END { print s+0 }’

awk '{
  for (i = 1; i <= NF; i++) {
    if ($i < 0) {
      $i = -$i;
    }
  }
  print
}'
```

```
awk '{ total = total + NF }; END { print total+0 }’

awk '$1 > max { max=$1; maxline=$0 }; END
   { print max, maxline }’

awk 'END { print $NF }’
awk 'NF > 4’
awk '$NF > 4'
```