

Tabular Data and AWK

Rupesh Nasre.
rupesh@cse.iitm.ac.in

Tabular Data

- Examples: Bills, inventory, marks, excel, csv, ...
- Array of structures
 - One structure == one record == one row
- Each column has a type.
- `awk` is designed to work with tabular data.
 - It needs to work across tables arbitrarily sized.

Roll Number	Name	City
7	Ananthram	Bangalore
17	Chanakya	Gottivalasa
34	Priya	Visakhapatnam
...	...	...

awk

- Powerful Unix tool for data processing
 - Full-fledged programming language
 - Works with patterns and actions
- Has an implicit loop (similar to `grep`)
- Created by Aho, Weinberger, Kernighan
 - See Aho's [3-minute video](#) on awk
 - Thought that it would be a throw-away tool
 - People quickly started using its first version.

Examples

- `awk '{print $1;}' input.csv` # space-separated
- `awk '{print $2, $1}' input.csv` # reorder columns
- `awk -F, '{print $0}' input` # comma-separated
- `awk '{print $2 "@" $1 " by " $3}' input` # generate descriptive string

Implement wc using awk.

- Recall wc.
 - `$ wc id-name-city.csv`
 - `95 250 2939 id-name-city.csv`
- We need the following facilities in awk:
 - Variables, initialization (once), printing once
- `awk 'BEGIN{...} {...} END{...}'`
- `awk 'BEGIN{x=0;} {x = x + 1;} END{print x;}'`
- In-built variables:
 - **NF** (number of fields)
 - **NR** (number of records)

Implement wc using awk.

- Recall wc.
 - `$ wc id-name-city.csv`
 - `95 250 2939 id-name-city.csv`
- `awk 'BEGIN{l=0; w=0;} {l = l + 1; w += NF;} END{print l, w}' id-name-city.csv`
 - `95 250`
- `awk 'BEGIN{l=0; w=0; c=0} {l = l + 1; w += NF; c+=length($0);} END{print l, w, c;}' id-name-city.csv`
 - `95 250 2844`
- `awk 'BEGIN{l=0; w=0; c=0} {l = l + 1; w += NF; c+=length($0) + 1;} END{print l, w, c;}' id-name-city.csv`
 - `95 250 2939`

Conditionals

- awk '{if (condition) statement}' input
- Example: Print every 4th city (id-name-city.csv).
 - awk '{if (\$1 % 4 == 0) print \$3;}' id-name-city.csv
- Example: Print names of low-attendance students (roll, name, marks, attendance).
 - awk '{if (\$4 < 25) print \$4, \$2;}' cs1234.txt | sort
 - awk '{if (\$4 < 25) print \$4, \$2;}' cs1234.txt | sort -n
 - awk 'BEGIN{roll=1; name=2; marks=3; attend=4;} {if (\$attend < 75) print \$attend, \$name;}' cs1234.txt | sort -n
 - awk 'BEGIN{name=2; attend=4;} {if (\$attend < 75) print \$attend, \$name; else ok++;} END{print ok;}' cs1234.txt
- Classwork: Find the sum of the last column.

Patterns

- `/pattern/ { action }`
- `awk '/Chennai/ {++count;} END{print count;}' id-name-city.csv`
- Classwork: How would you implement grep?
 - `awk '/pattern/ {print}' input`
- Print all Chennaites having roll number above 50.
 - `awk '/Chennai/ {if ($1 > 50) print}' input`
- Print the total size of all the files above 1000 bytes.
 - `ls -l | awk '{if ($5 > 1000) size += $5;} END{print size;}'`
 - `ls -l | awk '/Feb/ {if ($5>1000) sz+=$5;} END{print sz;}'`
- More complex patterns are possible using *regular expressions* (in a future course).

Map

- Associative array, key-value pairs
- e.g., `phone["Divyant"] = "7200762032";`
- We can use a map to count the number of residents of each city (as a summary).

Hyderabad 19

Surat 10

Kanpur 5

US 4

Bangalore 11

Kolkata 13

Visakhapatnam 10

Chennai 8

Dehradun 15

Map

- `awk -F, '{count[$3]++} END{for (city in count) print city, count[city];}' id-name-city.csv`
- Given rollno, name, grade, find the histogram of grade distribution.
- **Classwork:** Given rollno, name, marks, find the histogram.

awk script: with BEGIN and END

```
$ cat city1.awk
```

```
BEGIN{ print "Initialization"; }
```

```
{ print $1; }
```

```
END{ print "Final printing"; }
```

```
$ awk -F, -f city1.awk id-name-city.csv
```

```
Initialization
```

```
1
```

```
2
```

```
...
```

```
Final printing
```

awk script: with pattern

```
$ cat city2.awk
```

```
/Bangalore/ { print $2; }
```

```
$ awk -F, -f city2.awk id-name-city.csv
```

```
AKKASANI BINDHU MADHAVI
```

```
DHINESH GOMATHI SHANKAR
```

```
ARUNACHALAM
```

```
...
```

awk script: with conditionals

```
$ cat city3.awk
```

```
{ if ($1 > 20 && $1 < 30) { print $1, $2; } }
```

```
$ awk -F, -f city3.awk id-name-city.csv
```

```
21 DEV KAURAV
```

```
22 DHINESH GOMATHI SHANKAR ARUNACHALAM
```

```
...
```

```
28 GUDDITI HIMESH RAGHAVA CHANDRA
```

```
29 GUGULOTH SONIYA
```

awk script: with pattern and conditionals

```
$ cat city4.awk
```

```
/Bangalore/ {if ($1 > 20 && $1<30) {print $1, $2;}}
```

```
$ awk -F, -f city4.awk id-name-city.csv
```

```
22 DHINESH GOMATHI SHANKAR ARUNACHALAM
```

awk script: with multiple files

```
$ cat city5.awk
```

```
/Bangalore/ { print $1, $2; }
```

```
$ awk -F, -f city5.awk id-name-city.csv secondfile.csv
```

```
5 AKKASANI BINDHU MADHAVI
```

```
22 DHINESH GOMATHI SHANKAR ARUNACHALAM
```

```
...
```

```
82 SHREYAS SANGARSHAN S
```

```
92 VEDANTH SHRI GOVIND
```

```
CS25B010 Mithali Raj
```

awk script: NR and FNR

```
$ cat city6.awk
```

```
{ print NR, FNR, $1, $2; }
```

```
$ awk -F, -f city6.awk id-name-city.csv secondfile.csv  
thirdfile.csv
```

```
1 1 1 AARIT PANDA
```

```
2 2 2 AAYAN ALI KHAN
```

```
..
```

```
94 94 94 VIGHNESH SAWANT
```

```
95 95 95 YASH TULSHYAN
```

```
96 1 CS25B001 Yogesh Chavan
```

```
97 2 CS25B005 Kumar Gaurav
```

```
98 3 CS25B008 Virat Kohli
```

```
99 4 CS25B010 Mithali Raj
```

```
100 1 EE25B222 Suresh Bhat
```

```
101 2 EE25B333 Lata Mangeshkar
```

```
102 3 EE25B444 Sonu Nigam
```

Write a script to print file number at the start of its records.

awk script: NR and FNR

```
$ cat city7.awk
```

```
BEGIN{ print "----- This is the beginning." }
```

```
FNR == 1 { prevfilenum = filenum; ++filenum; }
```

```
prevfilenum != filenum { print "----- New file", filenum; }
```

```
{ print $3; prevfilenum = filenum; }    # main loop
```

```
END{ print "----- This is the end." }
```

```
$ awk -F, -f city7.awk id-name-city.csv secondfile.csv  
thirdfile.csv
```

```
// What is the output?
```

Homework: Try city8.awk and city9.awk.

Practice Problems

- Added to this document

In this course...

Application	Programming	Tools
Puzzle of grass, goat, lion	Pointers, structures	Editor, visualization
Data maintenance	File I/O	I/O redirection, make
Item grouping	Linked Lists	Debugging
Gene sequencing	Recursion, strings	Memory profiling, ulimit
Closest match	Matrices	Compute profiling, plotting
Keep the recent	Arrays	Version control
Cryptography	Strings, bitwise operators	Library (ncurses)
Unit testing	Command-line	Testing
Web-page creation, analysis	HTML, Strings	wget, shell scripting
Game	Graphics	Library (graphics)
Basics	Python	--

Our goal: *to have fun with automation*