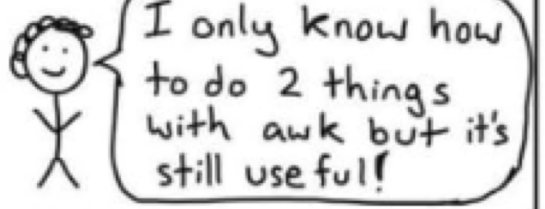
awk

awk

Julia Evans
@b0rk

awk is a tiny programming language for manipulating columns of data



basic awk program structure

```
BEGIN { ... }
CONDITION {action}
CONDITION {action}
END {...}
```

do action on lines matching CONDITION

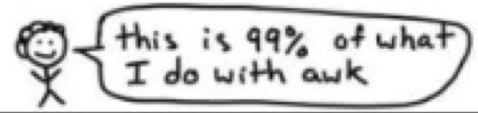
extract a column of text with awk

```
awk -F, '{print $5}'
```

column separator

single quotes!

print the 5th column



SO MANY unix commands print columns of text (ps! ls!)

so being able to get the column you want with awk is GREAT

A few more awk programs →

sum the numbers in the 3rd column

action

```
{s += $3}
END {print s}
```

at the end, print the sum!

print every line over 80 characters

```
length($0) > 80
```

condition

(there's an implicit {print} as the action)

awk

- Lee y ejecuta instrucciones por línea.
- Interpreta cada palabra/string delimitada por espacios, tabs, etc., como **fields**.
- Se puede usar tanto en la línea de comandos como en scripts.

Estructura

- En línea de comandos
  - **awk** '[instrucción]' archivo
- Para un script
  - **awk -f** script

Estructura

- **$0** toda la línea
- **$1** primer field
- **$2** segundo field

Ejemplo

$ awk '{print $1}' carsdata.csv

Make	Model	Year	Kms	Price
maruti	swift	2007	50000	5
honda	city	2005	60000	3
maruti	dezire	2009	3100	6
chevy	beat	2005	33000	2
honda	city	2010	33000	6
chevy	tavera	1999	10000	4
toyota	corolla	1995	95000	2
maruti	swift	2009	4100	5
maruti	esteem	1997	98000	1
ford	ikon	1995	80000	1
honda	accord	2000	60000	2
fiat	punto	2007	45000	3

```
user@jsgarc
Make
maruti
honda
maruti
chevy
honda
chevy
toyota
maruti
maruti
ford
honda
fiat
```

$awk '/honda/' carsdata.csv

```
honda   city    2005    60000   3
honda   city    2010    33000   6
honda   accord  2000    60000   2
```

$awk '/honda/ {print $2}' carsdata.csv

```
city
city
accord
```

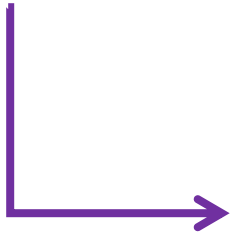
Fields

address.csv

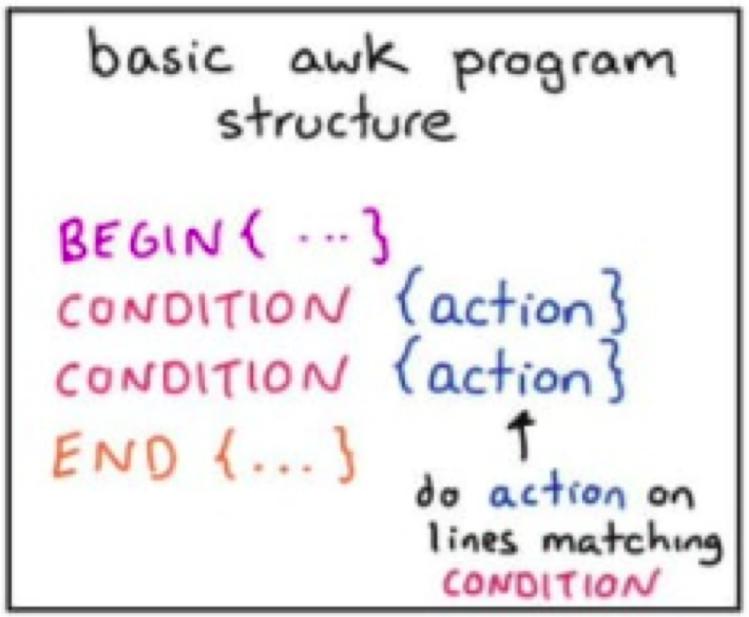
```
Massachusetts, John Daggett, 341 King Road, Plymouth, Massachusetts
Virginia, Alice Ford, 22 East Broadway, Richmond, Virginia
Oklahoma, Orville Thomas, 11345 Oak Bridge Road, Tulsa, Oklahoma
Pennsylvania, Terry Kalkas, 402 Lans Road, Beaver Falls, Pennsylvania
Massachusetts, Eric Adams, 20 Post Road, Sudbury, Massachusetts
Virginia, Hubert Sims, 328A Brook Road, Roanoke, Virginia
California, Amy Wilde, 334 Bayshore Pkwy, Mountain View, California
Massachusetts, Sal Carpenter, 73 6th Street, Boston, Massachusetts
```

```
John Daggett
Alice Ford
Orville Thomas
Terry Kalkas
Eric Adams
Hubert Sims
Amy Wilde
Sal Carpenter
```

$ awk -F “,” '{print $2}' address.csv



Field separator



BEGIN --- acciones antes de leer la primera línea de input

BEGIN { FS = "," } #FS -- field separator

END --- acciones después de leer todas las líneas de input

END { print FNR } #FNR -- número de línea actual

Condiciones

$5 ~ /MA/ { print $1 ", " $6 }

Variables

x = 1

x = $2

FS	Field separator
OFS	Output field separator
NR	Number of current input record
FNR	Number of current record relative to current input file
RS	Record separator
NF	Number of fileds

Count blank lines

```
/^$/ {
print x++
 }
```

average five grades

```
{ total = $2 + $3 + $4 + $5 + $6
avg = total / 5
print $1, avg }
```

```
$ ls -l
-rw-rw-rw-  1 dale     project   6041 Jan  1 12:31 com.tmp
-rwxrwxrwx   1 dale     project   1778 Jan  1 11:55 combine.idx
-rw-rw-rw-   1 dale     project   1446 Feb 15 22:32 dang
-rwxrwxrwx   1 dale     project   1202 Jan  2 23:06 format.idx
```

```
ls -l $* | awk '{print $5,"\t",$9}'
```

```
6041    com.tmp
1778    combine.idx
1446    dang
1202    format.idx
```

Estructura de Scripts

```
#!/bin/awk -f
BEGIN {
}
[instrucciones]
END {
}
```

Condiciones, loops y arrays

if (conditional **) statement [else statement]**

while (conditional **) statement**

for (expression ; conditional ; expression **) statement**

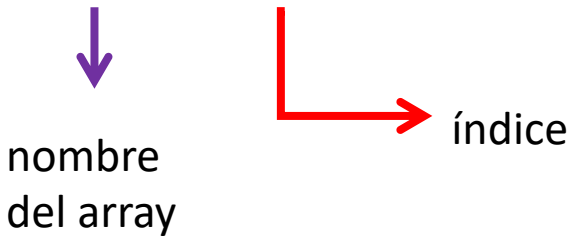
Condiciones, loops y arrays

```
'BEGIN{
FS = “,” }
{       if ( NF != 9 ){
                print $1, “incomplete data”}
        else {print $0}
}
END{print “Done”}'
```

```
'BEGIN { for(i=1;i<=6;i++) print "square of", i, "is",i*i; }'
```

Associative Arrays

array[$1] = valor




```
fruits["mango"] ="yellow";
fruits["orange"] ="orange";
```

```
print fruits["orange"] “\t” fruits["mango"]
```

```
orange yellow
```

a[$1] = $3”\t”$6

Associative array

```
#!/bin/awk -f
BEGIN{
        FS=”t”;OFS=”\t”}
        NR==FNR{ a[$1]=$2;next}
        {$7=a[$6];print}
END{print”DONE”}
```

Cambiar el RS y el FS

```
#!/bin/awk -f
{
        if ( $0 ~ /:/ ) {
                FS=":";
                $0=$0
        } else {
                FS=" ";
                $0=$0
        }
        #print the third field, whatever format
        print $3
}'
```

Cambiar el RS y el FS

```
#!/bin/awk -f
BEGIN {
# change the record separator from newline to
nothing
        RS=""
# change the field separator from whitespace to
newline
        FS="\n"
}
{
# print the second and third line of the file
        print $2, $3;
}
```