

# Introducció a l'awk

Albert Ruiz

Departament de Matemàtiques

6 de març de 2008



**Universitat Autònoma  
de Barcelona**

# Continguts

- 1 Introducció a l'awk
  - Algunes variables i funcions
  - Expressions regulars
  - En un fitxer

# El nom: awk

awk (a més de les inicials de Alfred Aho, Peter Weinberger i Brian Kernighan) és un llenguatge de programació pensat per a processar dades de fitxers de text. Aquest programa pot ser un fitxer sencer o bé també es pot entrar directament com a opció. Començarem veient alguns exemples a mode de línia de comanda.

## Exemple

Generem un fitxer amb un contingut per a fer proves:

```
$ ls -l /usr/bin > llistat.txt
```

Les opcions s'entren entre ', i s'agrupen entre claus { ... }.

Columnes: en general el símbol \$ seguit d'un número diu quina volem veure/tractar:

```
$ awk '{ print $7 }' llistat.txt
```

```
$ awk '{ print $1 " " $8 }' llistat.txt | nl (entre cometes hi ha el que volem veure literal).
```

# Algunes variables i funcions

## Variables i funcions

NR és el número de línia que es veu.

NF és el número de camps.

length dóna la llargada del seu argument.

split separa l'argument en trossos amb el separador escollit.

## Exemples

Si volem veure quants camps hi ha a cada línia:

```
$ awk '{ print NR " " $1 " " $8 " " NF }' llistat.txt
```

Veiem que hi ha algunes línies amb 10 camps.

Les variables també es poden utilitzar després del \$:

```
$ awk '{ print NR " " $1 " " $NF }' llistat.txt
```

```
$ awk '{ print $8, length($8)}' llistat.txt | sort  
-k2,2n
```

# Més exemples

## Exemples

Comprovem quina és la primera columna del fitxer `/etc/passwd`:

```
$ awk '{print $1}' /etc/passwd
```

Si volem separar pel caràcter `:`

```
$ awk -F":" '{print $1}' /etc/passwd
```

O bé

```
$ awk '{split($1,tros,":"); print tros[1]}'  
/etc/passwd
```

# Més exemples

## Més exemples

L'expressió següent fa el mateix que la comanda `wc`:

```
$ awk 'BEGIN {c=0;p=0;} {c=c+length($0);p=p+NF;} END  
{print c,p,NR}' llistat.txt
```

L'expressió següent ordena un fitxer amb una columna de números (el creem abans), calcula la diferència entre dos i l'anterior i torna a ordenar el llistat final.

```
$ awk '{print $5}' llistat.txt > num.txt  
$ sort -n num.txt | awk 'BEGIN{ol=0;}  
{printf("%d\n",$1-ol); ol=$1;}' | sort -n
```

# Més exemples

## Més exemples

Si volem aplicar la mateixa instrucció a tots els fitxers (o un conjunt de fitxers) d'un directori podem utilitzar l'awk com a pipa de l'ls.

Per exemple, creem una còpia de tots els arxius del directori afegint al davant copia\_:

```
$ ls | awk '{print "cp "$1"copia_"$1}' | sh
```

Si volem fer una còpia de tots els jpg d'un directori a un subdirectori anomenat per\_enviar refent el tamany a 800 × 600 fem:

```
$ ls *.jpg | awk '{print "convert -size 800x600 "$1"-resize 800x600 per_enviar/"$1}' | sh
```

# Expressions regulars

## Expressions regulars

També podem filtrar les dades per expressions regulars. Per a això hem d'escriure l'expressió regular entre /.

## Exemples

Si volem veure què executen els enllaços de /usr/bin:

```
$ awk '/^l/ {print NR $8$9$10 NF}' llistat.txt
```

Agafem el fitxer awk.tar.gz de

```
http://mat.uab.cat/~albert/curs1.
```

```
$ wget http://mat.uab.cat/~albert/curs1/awk.tar.gz
```

El descomprimim i mirem el que fan les execucions següents.

```
$ awk '/^Lleó/ {print $2 $3}' observ.txt
```

```
$ awk '{print $1 >> "primera.txt"; print $2 >> "segona.txt"}' observ.txt
```



# En un fitxer

## Comandes en un fitxer

L'awk permet escriure programes més complexos.

## Exemples

Per a veure més opcions de l'awk mirem el fitxer `filtre.awk`:

```
$ less filtre.awk
```

Per a utilitzar-lo amb l'awk executem:

```
$ awk -f filtre.awk observ.txt
```