Lab. di Sistemi Operativi

"Sed e Awk"

Sommario

- Comandi di ricerca:
  - Sed (Stream Editor)
    - Selezione di un range di righe
    - Ricerca e sostituzione
  - AWK

- Stream editor: sed -

Esercizio n° 1

- Utilizzando il comando **sed**, stampare su video le prime **cinque** righe del file /etc/passwd

  Esempio: **sed '[address1[, address2]]d** ' (selezione di un range di righe)

Soluzione

```
sed  '6,$d' /etc/passwd
```

d è il comando di cancellazione. In questo caso incomincia ad eliminare tutte le righe a partire dalla sesta fino all' ultima riga contenuta nel file ($ sta per ultima riga del file)

Esercizio n° 2

- Supponendo di avere un file **gruppiLSO.txt** con la lista dei gruppi e dei relativi punteggi formattato usando ";" come separatore di campo.

Esempio di file:

NomeGruppo;Punteggio;matricola1;……;matricolaN
lso01InfoNa03;69,00;566100;…;566101
lso01InfoNa08;70,00;566300;…;566504
lso01InfoNa21;65,33;566500;…;566601
lso01InfoNa15;63,56;566600,…;566432
……;……;……;……;
……;……;……;……;

Esercizio n° 2

- **Sostituire** utilizzando il comando **sed** **tutti** i separatori di campo ";" con caratteri di spazio " " in modo tale da avere il seguente output:

```
NomeGruppo     Punteggio
lso01InfoNa03  69,00
lso01InfoNa08  70,00
lso01InfoNa21  65,33
lso01InfoNa15  63,56
……..……..   ……..…..
……..……..   ……..…..
```

Soluzione Esercizio n° 2

Utilizzo delle quattro parti del comando sostituzione (s)

sed 's/regexp/replacement/flags' <nome_file>

Soluzione

```
sed 's/;/ /g' gruppiLSO.txt
```

oppure

```
cat gruppiLSO.txt | sed 's/;/ /g'
```

Esercizio n° 2'

- Utilizzando il comando **sed**, nel file /etc/passwd per gli utenti che usano bash sostituire il contenuto del campo password con "password"

Esempio:

root:x:0:0:root:/home/root:/bin/bash

root:password:0:0:root:/home/root:/bin/bash

Soluzione

```
sed '/bash/s/:x:/:password:/g' /etc/passwd
```

*flag **g**: applica la sostituzione con il carattere di spazio a tutte le occorenze ":"*

Esercizio n° 2"

- Nel file **gruppiLSO.txt** sostituire utilizzando il comando **sed** tutti i separatori di campo ";" con caratteri di spazio " " e commentare con # tutti i gruppi che contengono matricole 566-xxx

Soluzione

```
sed –e '/566/s/^/#/g' –e 's/;/ /g' gruppiLSO.txt
```

Un modo di combinare più comandi è quello di usare -e prima di ciascun comando

Esercizio n° 3

- Nel file **gruppiLSO.txt** utilizzando il comando **sed** nelle prime 3 righe sostituire tutte le matricole 566-xxx con 566, dalla 4 riga cancellare le matricole 566-xxx

Soluzione

```
sed -e‘1,3s/566.*;/566;/g’-e ‘4,$s/566.*;/;/g’
```

- AWK -

- Esempi d'uso -

- AWK: ha bisogno di **due elementi** per funzionare:
  - il flusso in input (file_testo)
  - e il codice per elaborarlo (codice_awk).

Quindi:

```
awk -opzioni '{ codice_awk }' file_testo
```

Oppure

```
cat file_testo | awk -opzioni '{ codice_awk }'
```

Invocazioni

1. `awk` *`programma file-dati`*

   `awk '{ print "hello" }' pippo.txt`

   Esegue il programma sul contenuto del file

1. `awk` *`programma`*

   Esegue il programma su *stdin*

1. `awk -f` *`file-programma file-dati`*
2. `awk -f` *`file-programma`*

   Esegue il programma contenuto nel primo file sul contenuto del secondo file (o su *stdin*)

Sintassi di un Programma

Sequenza di *regole*:

[pattern] [azione]

- Pattern:
  - espressione booleana a >= 1
  - espressione regolare /ciao/
- Azione:
  - blocco di istruzioni { a = b + 1; print $0 }

Semantica

Sequenza di *regole*:

[pattern] [azione]

1. Divide l’input in righe
2. Per ogni riga valuta tutte le regole in ordine
3. Esegue tutte le azioni il cui pattern risulta “vero”

Esempio

```
awk '/a/ { print "trovato" }'
```

Stampa “trovato” per ogni riga che contiene almeno una “a”

Variabili Speciali

- $0 riga corrente
- $1…$9 colonna i-esima della riga corrente
- NF numero di colonne nella riga corrente *(number of fields)*
- NR numero della riga corrente

Variabili modificabili:

- FS separatore di colonna *(field separator)*
- RS separatore di riga *(row separator)*

Cambiare il Separatore di Colonna

- Default: spazi e tabulazioni
- Cambiare separatore internamente: `FS=":"`
- Cambiare separatore esternamente:

```
awk -F<separatore> ...
```

- Esempio:

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

Esempio

```
awk '$1 ~ /^a/ { print $2 }'
```

Stampa la seconda colonna delle righe la cui prima colonna comincia per “a”

Abbreviazioni

Sequenza di regole:

[pattern] [azione]

- Azione senza pattern — eseguita su tutte le righe
- Pattern senza azione — stampa il rigo
- Pattern “/regexp/” — abbrev. di “$0 ~ /regexp/”
- Azione “print” — abbrev. di “print $0”

Esempio

Tutti questi sono equivalenti:

```
awk ‘{n++}  n>10 { print $0 }’
awk ‘{n++}  n>10 { print }’
awk ‘{n++}  n>10’
awk ‘NR>10’
```

Stampano tutto, a cominciare dalla riga 11

Esercizio n° 4

- Applicare la lista dei gruppi ottenuta nell' Esercizio N° 2 , al comando **awk** che con il comando **sort** deve produrre in output una lista dei gruppi in ordine decrescente di punteggio:

Output:

Punteggio	*NomeGruppo*
70,00	lso01InfoNa08
69,00	lso01InfoNa03
65,33	lso01InfoNa21
63,56	lso01InfoNa15

Soluzione Esercizio n° 4

cat gruppiLSO.txt | sed –e 's/:/ /g' | awk '{print $2 " " $1}' | sort –nr

output

lso01InfoNa03 69,00
lso01InfoNa08 70,00
lso01InfoNa21 65,33
lso01InfoNa15 63,56

input

awk '{print $2 " " $1}'

output

69,00 lso01InfoNa03
70,00 lso01InfoNa08
65,33 lso01InfoNa21
63,56 lso01InfoNa15

input

sort -nr

output

70,00 lso01InfoNa08
69,00 lso01InfoNa03
65,33 lso01InfoNa21
63,56 lso01InfoNa15

Passare variabili di shell ad awk

- Supponiamo a=7 (variabile bash)
- Per ogni rigo, vogliamo stampare il valore di “a” e poi il primo campo del rigo
  - `awk “{ print $a $1 }”` equivale a `awk “{ print 7 }”`
  - `awk '{ print $a $1 }'` equivale a `awk '{ print $0 $1 }'`

Come si fa?

```
awk -v mia_a=”$a” '{ print mia_a $1 }'
```

Pattern Speciali

- **BEGIN**: esegue l'azione prima di leggere il primo rigo
- **END**: esegue l'azione dopo aver letto l'ultimo rigo

Esempio:

```
awk  'BEGIN { print “Inizio” }
            { print $1 }
      END   { print “Fine” }'
```

Esempio

Stampare la dimensione totale di tutti i file della directory corrente con estensione *.txt*

```
ls -l | awk '/.txt$/ { tot += $5 }
            END      { print tot }'
```

Esercizi

1. Stampare solo le righe dispari
2. Stampare solo il numero di righe che contengono almeno una ‘x’
3. Stampare il nome del file più grande della directory corrente

Esercizio

Convertire in HTML l'elenco della directory corrente

Esempio:

Se la directory contiene i seguenti file:

```
> ls
a.txt b.txt prova.c
```

l'output del programma awk potrebbe essere il seguente:

```
> ls | awk -f convert-to-HTML.awk
<HTML><BODY>
<ul>
<li>a.txt</li>
<li>b.txt</li>
<li>prova.c</li>
</ul>
</BODY></HTML>
```

Esercizio n° 5

- Ripetere l' Esercizio N° 4 senza utilizzare il comando **sed** (solo **awk** e **sort**) partendo quindi da un file **gruppiLSO.txt** avente il seguente contenuto:

NomeGruppo; Punteggio
lso01InfoNa03;69,00
lso01InfoNa08;70,00
lso01InfoNa21;65,33
lso01InfoNa15;63,56
……;……
……;……

Soluzione Esercizio n° 5

awk –F';' '{print $2 " " $1}' gruppiLSO.txt | sort –nr

output

69,00 lso01InfoNa03
70,00 lso01InfoNa08
65,33 lso01InfoNa21
63,56 lso01InfoNa15

input

sort -nr

output

70,00 lso01InfoNa08
69,00 lso01InfoNa03
65,33 lso01InfoNa21
63,56 lso01InfoNa15

Esercizio n° 6

- Facendo uso del comando **awk**, elencare gli utenti diversi che usano "**bash**" come shell di default presenti nel file /etc/passwd

Esempio:

root:x:0:0:root:/home/root:/bin/bash

Soluzione

cat /etc/passwd | awk 'BEGIN {FS=":"} ($7==“/bin/bash”) {print $1}'

Esercizio n° 7

- Facendo uso del comando awk, elencare tutti i pid e gli username dei processi con CPU% maggiore di 0.1. Stampare quindi l'utilizzo totale di CPU da parte di tali processi

Soluzione

```
ps -aux | awk'{if($3 > 0.1){print $3;tot+=$3}}
END {print "totale:" tot }'
```

Esercizio n° 8

- Facendo uso del comando awk, elencare tutti i pid e ppid dei processi dell'attuale user. Stampare quindi il numero totale di processi

Soluzione

```
ps u $USER | awk'{print $1 $2; tot+=1} } END
{print "totale:" tot }'
```

- Fine Esercitazione -