Soluzioni ad Alcuni Esercizi sui comandi UNIX

Ecco un elenco di problemi che dovete risolvere usando i comandi Unix. Per ogni esercizio, la soluzione consiste nel trovare il comando o la combinazione di comandi (in questo caso inseriti in un shell script) che risolvono il problema. Non usate programmi C.

(NOTA: gli esercizi marcati con “*” sono più difficili, poichè richiedono l’uso di costrutti/concetti non ancora visti a lezione)

1

- 1) Visualizzare le prime n righe di un file di testo

```
$> head -n nome-file
```

- 2) Visualizzare le ultime n righe di un file di testo

```
$> tail -n nome-file
```

- 3) Ordinare le righe di un file di testo (anche in senso inverso)

```
$> sort nomefile
```

```
$> sort -r nomefile
```

```
$> sort -r nomefile -o nomefile
```

2

- 4) Ordinare le righe di un file in base al terzo campo del file (si supponga che i vari campi del file siano separati dallo spazio. Una riga del file ha quindi la forma:

```
Mario Rossi 0116704543 mrrss102030)
```

```
$> sort -k 3,3 nomefile
```

oppure:

```
$> sort +2 -3 nomefile
```

- 5) Modificare un file di testo rimuovendone le prime n righe. Potreste dover usare un file temporaneo?

```
$> wc -l nomefile (sia  $m$  il numero restituito)
```

```
$> tail -k nomefile > tmp (dove  $k = m - n$ )
```

```
$> mv tmp nomefile
```

3

- 6) Costruire un shell-script che risolva l'esercizio 5) in modo parametrico, ricevendo cioè in input il nome del file su cui operare e il numero di righe da rimuovere. (*)

```
len=`cat $1 | wc -l`  
resto=`expr $len - $2`  
tail -$resto $1 > tmp  
mv tmp $1
```

4

- 7) Suddividere un file di 50 righe di testo in 5 file contenenti ciascuno 10 righe del file iniziale

```
$>split -10 nomefile
```

- 8) rimuovere tutti i file con un certo nome (es: *.c) SOLO se si trovano in cartelle 2 livelli sotto quella corrente

```
$> rm */*/*.c
```

- 9) Rimuovere tutti i file con un certo nome (es: *.c) a partire dalla directory corrente (*)

```
$> find . -name "*.c" -exec rm {} \;
```

5

- 10) Modificare la priorit di un processo (*)

```
$> renice 10 -p PID
```

- alza di 10 la priorit di base (il *nice*) del processo con process_id PID.
- Per vedere la priorit modificata di un processo dovete usare l'opzione -l del comando ps, e leggere il valore riportato alla colonna NI (NICE) (Attenzione a non confondervi con la colonna PRI, che e' il valore della priorit corrente dei processi, che viene ricalcolata periodicamente dal sistema operativo in base a vari parametri, tra cui la quantita' di CPU usata fino a quel momento dal processo relativo.

6

- 11) Stampare in sequenza due volte 3 stringhe passate in input. (*) Ad esempio:

```
$> ristampa qui quo qua  
    qui quo qua  
    qui quo qua
```

in uno script shell:

```
echo $1 $2 $3  
echo $1 $2 $3
```

- Nota: avere provato usando il comando "printf" ?

7

- 12) Visualizzare il nome del terminale su cui si sta lavorando

```
$> tty
```

- 13) Concatenare 2 file passati in input in un terzo file

```
$> cat fileA fileB > fileC
```

8

- 14) Contare quanti processi sulla macchina NON sono associati ad un terminale

```
$> ps agx |grep -v pts |grep -v console | wc -l
```

- (nota: il primo grep elimina dalla lista tutti i processi associati ad un terminale virtuale --- cioe' una finestra shell --- il secondo grep elimina i processi associati all'unico terminale fisico della macchina, indicato nelle macchine unix col termine "console")

9

- 15) aggiungere al fondo di un file A il contenuto di un altro file B SENZA usare file temporanei. (*)

```
$> cat fileA >> fileB
```

- (nota: mentre il carattere di redirezione > sovrascrive il file di output fileB, >> appende il contenuto di fileA alla fine di fileB)

10

- 16) lanciare un comando ad un ora prestabilita

- Se il o i comandi da eseguire sono contenuti nel file "command-file" si puo' usare:

```
$> at -f command-file -t time
```

- dove il formato di *time* (che permette di specificare anche un'eventuale data) e' del tipo: MMDDHHMM (mese, giorno, ora, minuti)

11

- Se invece si vuole specificare direttamente il programma/comando da eseguire:

```
$> at -t time  
at> command-name  
<EOT>
```

- Digitando la prima riga del comando si entra nell'ambiente interattivo del comando at. A questo punto si puo' digitare il nome del comando/programma da eseguire seguito da <Enter>. Si possono specificare più comandi/programmi uno dopo l'altro. Al termine si puo' uscire dall'ambiente interattivo digitando ctrl-D (<EOT>).

12