Esempi shell

Esempio 1

- Creare uno script (file comandi) che abbia la sintassi:

./ps_monitor.sh [N]

- Lo script:
  - in caso di assenza dell'argomento, deve mostrare i processi di tutti gli utenti (compresi quelli senza terminale di controllo) con anche le informazioni sul nome utente e ora di inizio;
  - Se viene passato come argomento un intero (N) deve mostrare i primi N processi

Suggerimento: usare il comando **ps**

NB: non tutte le righe prodotte in output da ps hanno contenuto informativo rilevante

Soluzione

```
#!/bin/bash
if [ $# -eq 0 ]  # se non ci sono parametri
then
    ps aux
else
    ps aux | head -n `expr $1 + 1`
#consideriamo che c'e' anche una riga di intest.
fi
```

Esempio 2

- Creare uno script che abbia la sintassi

  ./lines_counter.sh <directory> [up|down]

- Lo script deve elencare i file contenuti nella directory con relativo numero di linee, ordinati in senso crescente (up) o decrescente(down)

NOTA: controllare:

–Che il primo argomento sia effettivamente una directory

–Che il secondo argomento sia la stringa up o down

Soluzione

```
#!/bin/bash
if [ $# -ne 2 ] #sintassi sbagliata
then
  echo ”SINTASSI: lines_counter.sh <directory> [up|down]"
  exit 1 #uscita anomala
fi
if [ -d $1 ] #vero se $1 è una directory
then
    cd $1
    if [ $2 = "up" ]
    then
      cat * | wc -l | sort -n

#1. viene espansa la lista di tutti i file presenti in $1
#2. su ogni elemento viene eseguito il conteggio
#3. viene effettuato l'ordinamento sui conteggi
```

```
elif [ $2 = "down" ]
    then
     cat * | wc –l | sort -nr #l'ordinamento è inverso
    else
       echo "ERROR: 'up' or 'down'"
       exit 2 #uscita anomala
    fi
 fi
else
 echo "$1 should be an existent directory"
 exit 2 #uscita anomala
fi
```

Esempio 3

- Creare uno script che abbia la sintassi

./backup.sh <nomefile> <nomebackup>

- Se il file è una directory, lo script deve:
  - creare una sottodirectory(rispetto a livello corrente) di nome: <nomefile>_<nomebackup>
  - copiare ricorsivamente in essa il contenuto della directory
- Se il file è un file normale, lo script deve crearne 5 copie di nome <nomefile>*i<nomebackup> i=1..5

Soluzione

```
#!/bin/bash

#IPOTESI: considero solo file e direttori nel dir.corrente

if [ $# -ne 2 ]
then
    echo "USAGE: backup.sh <filename> <backupstring>"
    exit 1
fi

if [ -d $1 ]
#-restituisce 1 se il primo parametro e' una directory
then
   cp -R $1 "$1_$2"
```

```
elif [ -f $1 ] #controlla che $1 sia un file normale
then
    for i in 1 2 3 4 5 #i cicla sugli el. della lista
    do
      cp $1 "$1*$i$2"
#i doppi apici inibiscono l'espansione di * ma non di $
    done
else
    echo "$1 should be a valid directory or file"
fi
```

Esercizio shell
(esame 12 luglio 2010)

Si realizzi un file comandi Unix con la seguente interfaccia:

`copy.sh <dir> <string> <dest>`

- <dir> e <dest> direttori assoluti esistenti nel filesystem;
- <string> una stringa.

Dopo aver effettuato tutti gli opportuni controlli sui parametri in ingresso, il file comandi si deve occupare di cercare, in ciascun sottodirettorio di dir, tutti i **file regolari nelle cui prime 10 righe compaia <string> almeno una volta**. Per ciascun file così trovato all'interno di un sottodirettorio, si copi tale file in un opportuno sottodirettorio di <dest> del tipo **`<dest>/N`** **(**cioè un sottodirettorio di <dest>) il cui nome sia uguale al **numero effettivo** di occorrenze di <string> trovate nelle prime 10 righe del file.

Ad esempio, supponendo di invocare il comando con

copy.sh /home/user pdf /home/backup

e di avere la seguente condizione su filesystem:

/home/user/prova.txt (3 occ. di pdf nelle prime 10 righe)
/home/user/prova1.txt
/home/user/prova.xml (7 occ. di pdf nelle prime 10 righe)
/home/user/dir1/prova.txt
/home/user/dir1/prova.pdf

il file comandi creerà e riempirà il direttorio di backup in questo modo:

/home/backup/7/prova.xml
/home/backup/3/prova.txt

Esempio di Soluzione

- Esplorazione di una gerarchia nel file system: imposto la soluzione in modo ricorsivo:
  - un file comandi può invocare se stesso: `$0 ..`
  - In caso di ricorsione, lo script chiamante attende il completamento dell'esecuzione dell'invocazione ricorsiva

- 2 file:
  - **copy.sh:** controllo argomenti, settaggio path e invocazione del file ricorsivo:
  - **copy_rec.sh:**
    - **Esecuzione ricorsiva a partire dalla radice della gerarchia**

Copy.sh

```
#!/bin/sh
# Controllo parametri
if test $# -ne 3
then
   echo "usage:$0 <dir> <string> <dest>"
   exit 1
Fi
case $1 in
   /*) ;;
   *)  echo "$1 is not an absolute directory"
       exit 4;;
esac
if ! test -d "$1"
then
   echo "$1 is not a valid directory"
   exit 5
fi
```

```
#..continua
case $3 in
   /*) ;;
   *)  echo "$3 is not an absolute directory"
exit 4;;
esac
if ! test -d "$3"
then
   echo "$3 is not a valid directory"
   exit 5
fi

# Invocazione script ricorsivo:
oldpath=$PATH
PATH=$PATH:`pwd`
copy_rec.sh "$1" $2 "$3"
PATH=$oldpath
```

Copy_rec.sh

```
#!/bin/sh
cd "$1”
for f in *
do
    if test -d "$f"
    then
         $0 "$f" $2 "$3"
    elif test -f "$f"
    then
        count=`head –n 10 "$f" | grep -o "$2" | wc -l`
        if test $count -gt 0
        then
            if ! test -d "$3"/$count
            then
                mkdir "$3"/$count
            fi
            cp "$f" "$3"/$count
        fi
    fi
done
```