Fondamenti di Informatica e Laboratorio T-AB
Ingegneria Elettronica e Telecomunicazioni

Lab Shell

Scelta della shell

La prima riga di un file comandi deve specificare ***quale shell si vuole utilizzare***: **`#! <shell voluta>`**

- Es: **`#!/bin/bash`**
- # è visto dalla shell come un commento ma…
- #! è visto da SO come identificatore di un file di script

  SO capisce così che l'interprete per questo script sarà **`/bin/bash`**

- Se questa riga è assente viene scelta la shell di preferenza dell'utente

Passaggio parametri

`./nomefilecomandi arg1 arg2 ... argN`

Gli argomenti sono ***variabili posizionali*** nella linea di invocazione contenute nell'ambiente della shell

- **$0** rappresenta il comando stesso
- **$1** rappresenta il primo argomento ...

- è possibile far scorrere tutti gli argomenti verso sinistra

  ➔ **`shift`**

  $0 non va perso, solo gli altri sono spostati ($1 perso)

	$0	$1	$2
prima di **shift**	DIR	-w	/usr/bin
dopo **shift**	DIR	/usr/bin	

- è possibile riassegnare gli argomenti ➔ **`set`**
  - **`set exp1 exp2 exp3 ...`**
  - gli argomenti sono assegnati secondo la posizione

Altre informazioni utili

Oltre agli argomenti di invocazione del comando

- **`$*`** insieme di ***tutte le variabili posizionali***, che corrispondono arg del comando: $1, $2, ecc.
- **`$#`** ***numero di argomenti*** passati (***$0 escluso***)
- **`$?`** valore (int) restituito dall'ultimo comando eseguito
- **`$$`** id numerico del processo in esecuzione (pid)

Semplici forme di input/output

- **`read var1 var2 var3`** **`#input`**
- **`echo var1 vale $var1 e var2 $var2`** **`#output`**
  - **read** la stringa in ingresso viene attribuita alla/e variabile/i secondo corrispondenza posizionale

Comandi da usare

- cat
  - Concatena files e li stampa a video
- test
  - Verifica una condizione
- sort
  - Ordina le righe passate in input
- grep
  - Cerca testo nei files
- chmod
  - Modifica diritti (rwx) sui files

USARE man PER I DETTAGLI

Redirezione I/O

Operatori binari

- |
  - l'output del commando di sx diventa input del commando di dx
- > o >>
  - L'output del commando di sx viene scritto (o appeso) nel file a dx
- <
  - Il contenuto del file di dx diventa l'input del commando di sx

Esercizio 0

Creare ed eseguire un primo script shell utilizzando cat e chmod

```
_:cat > ilmioprimoscript
#!/bin/bash
echo $1
^C
_:chmod 0777 ilmioprimoscript
_:./ilmioprimoscript argomento1
argomento1
```

A questo punto aprire il file con l'editor preferito e sperimentare la sintassi base della shell

```
_:mousepad ilmioprimoscript
```

Strutture di controllo: alternativa

```
if <lista-comandi>
  then
     <comandi>
  [elif <lista_comandi>
   then <comandi>]
  [else <comandi>]
  fi
```

ATTENZIONE:

- le parole chiave (do, then, fi, ...) devono essere o ***a capo o dopo il separatore ;***
- if controlla il valore in uscita ***dall'ultimo comando di <lista-comandi>***

Esercizio 1

Analizzare il seguente script e spiegarne il funzionamento (eventualmente provandolo)

```
#!/bin/bash
# Basic if statement
if [ $1 -gt 100 ]
then
echo Hey that\'s a large number.
pwd
fi
date
```

Alternativa multipla

```
# alternativa multipla sul valore di var
case <var> in
   <pattern-1>)
       <comandi>;;
   ...
   <pattern-i> | <pattern-j> | <pattern-k>)
       <comandi>;;
   ...
   <pattern-n>)
       <comandi> ;;
esac
```

Importante: nell'alternativa multipla si possono usare metacaratteri per fare pattern-matching (non sono i "soliti" metacaratteri su nome di file)

Esercizio 2

Analizzare il seguente script e spiegarne il funzionamento (eventualmente provandolo)

```
#!/bin/bash
# case example
case $1 in
start)
echo starting;;
stop)
echo stoping;;
restart)
echo restarting;;
*)
echo don\'t know;;
esac
```

Cicli enumerativi

```
for <var> [in <list>] # list=lista di stringhe
do
   <comandi>
done
```

- scansione della lista <list> e ***ripetizione del ciclo per ogni stringa presente nella lista***
- scrivendo solo `for i` si itera con valori di **i in $***

Esercizio 3

Analizzare il seguente script e spiegarne il funzionamento (eventualmente provandolo)

```
#!/bin/bash
# Basic for loop
names=`Syd Richard Nick David Roger'
for name in $names
do
echo $name
done
echo All done
```

Ripetizioni non enumerative

```
while <lista-comandi>
do
   <comandi>
done
```

Si ripete per tutto il tempo che il valore di stato dell'ultimo comando della lista è zero (successo)

```
until <lista-comandi>
do
   <comandi>
done
```

Come while, ma inverte la condizione

Uscite anomale

- vedi C: **continue**, **break** e **return**
- **exit** [**status**]: system call di UNIX, anche comando di shell

Esercizio 3

Analizzare il seguente script e spiegarne il funzionamento (eventualmente provandolo)

```
#!/bin/bash
# Basic while loop
counter=1
while [ $counter -le 10 ]
do
echo $counter
((counter++))
done
echo All done
```

Esercizio 4

Produrre uno script shell che

- Se riceve un numero di parametri minore di 2 restituisce un messaggio di errore e termina
- Se il primo parametro non è uno fra s e ps restituisce un messaggio di errore e termina
- Se il primo parametro è s: cerca in tutti i file indicati dal secondo parametro in poi le righe che contengono 'scanf' e le stampa
- Se il primo parametro è ps conta separatamente tutte le righe che contengono 'printf' e 'scanf in tutti i file indicati dal secondo parametro in Avanti e stampa a video un messaggio che indica se sono più le printf o le scanf

Esercizio 4 - Soluzione

```
#!/bin/bash
if test $# -lt 2
then
   echo Errore: almeno 2 argomenti;
   exit;
fi

case $1 in
s)
   shift #Escludiamo il primo argomento passato ('s')
   grep scanf $@;; #$@ eqivale a tutti gli argomenti
```

Continua pagina seguente…

Esercizio 4 - Soluzione

…segue da pag precedente

```
ps)
    shift #Escludiamo il primo argomento passato ('ps')
    grep scanf $@ | wc -l > tempfile
    read nscanf < tempfile
    grep printf $@ | wc -l > tempfile
    read nprintf < tempfile
    if [ $nscanf -gt $nprintf ]
    then
          echo Più scanf
    elif [ $nscanf -lt $nprintf ]
          echo Più printf
    else
          echo Stesso numero di scanf e printf
    fi;;
*)
    echo Errore: il primo argomento deve essere s o ps;
    exit;
esac
```