Il costrutto if

- if condizione
  then
  azione1
  azione2
  ...
  fi
- if condizione
  then
  azione1
  azione2
  ...
  else
  azione3
  azione4
  ...
  fi

if condizione
then
azione1
azione2
...
elif condizione
then
azione3
azione4
...
else
azione5
azione6
...
fi

Condizioni per l'if

- Non esistono variabili booleane
- Ci sono vari modi di specificare la condizione, ognuno con le sue particolarità e "osticità"
- Gli usi "soliti" dell'if sono alquanto antiintuitivi:
  - se si applica if direttamente ad un comando, la condizione è soddisfatta se il comando ritorna zero, non è soddisfatta se ritorna un valore diverso da zero
  - se si applica if ad una variabile, la condizione è sempre soddisfatta purchè la variabile sia definita

- Poichè è facile confondersi meglio rinunciare agli usi soliti e ricorrere agli usi tipici della shell

Costrutto test

- `if [ qualcosa ]` è una forma abbreviata equivalente a `if test qualcosa` (necessari spazi vicino a parentesi !!)
- Facendo man test si può vedere l'insieme di test a disposizione (di nuovo non sempre intuitivi):
  - confronti tra stringhe: `S1 == S2, S1!= S2` (da non usare per interi, obbligatori gli spazi prima e dopo l'operatore di confronto !!)
  - test lunghezza stringa: `-n S1` (lunghezza > 0), `-z S1` (lunghezza pari a 0)
  - confronti tra interi: `INT1 -eq INT2` (o `-ge,-gt,-lt,-le,-ne`)
  - confronti tra file: `FILE1 -nt FILE2` (FILE1 più recente di FILE2) e viceversa `FILE1 -ot FILE2`; di uso particolare `FILE1 -ef FILE2`
  - test proprietà di un file: `-e FILE` (esistenza), `-r FILE` (esiste ed è leggibile), e così via (`-b,-c,-d,-f,-g,-h,-G,-k,-L,-O,-p,-s,-S,-u,-w,-x`) (vedi anche http://tldp.org/LDP/abs/html/fto.html)
- I connettivi logici sono il solito `!` per il not, `-a` tra due espressioni per indicare and, `-o` per indicare or

Esempio di uso di condizioni composte

- Se uno script accetta solo uno o due argomenti si può mettere all'inizio un controllo di questo tipo

```
if [ $# -eq 0 -o $# -gt 2 ]
 then
  echo "Numero di parametri errato"
  exit 1
fi
```

- Come in ogni attività di programmazione un controllo anche minimo sugli argomenti passati è molto consigliabile

Altre condizioni

- Oltre alle condizioni fornite dal costrutto test (parentesi quadre singole), si possono usare all'interno di un if:

  - doppie parentesi quadre `[[ ... ]]`: analoghe alle parentesi quadre singole ma ammettono l'uso di `&&` e `||` per and e or

  - doppie parentesi tonde: `(( ... ))` sono un costrutto generico per operazioni aritmetiche, di uso non frequente e non immediato, consultare documentazione prima di avventurarsi

Il case

```
case "$variable" in

stringa1 ) #stringa1 può essere il valore di una variabile p.e. "$varcaso1"
command...
;;

stringa2 ) #stringa2 può essere il valore di una variabile p.e. "$varcaso2"
command...
;;
*)
default command
;;
esac
```

Esempi articolati su: http://tldp.org/LDP/abs/html/testbranch.html

Un esempio

```
case "$1" in
  "E" | "e" )
  # Esegui qualcosa corrispondente alla lettera e/E
  ;;
  "J" | "j" )
  # Esegui qualcosa corrispondente alla lettera j/J
 ;;
 * )
  # Tratta il caso di default
;;

esac
```

Ciclo for

```
for variabile in lista
do
  comando1
  comando2
  ....
done
```

- Come solito, il corpo del ciclo viene ripetuto facendo assumere successivamente a `variabile` tutti i possibili valori presenti in `lista`
- `do` e `done` sono comandi: devono stare su linee separate

Ciclo for

- La lista può essere semplicemente specificata come una sequenza di valori separati da spazi (o altri caratteri spaziatori):
  `for nomedocente in Andrea Nicola PietroB PietroM`
- La lista può anche essere ottenuta per filename expansion
  `for file in *.txt`
- La lista può anche essere ottenuta come risultato di un comando

```
for nome in $(cat elenco.txt)
   ...
for file in $(find . -name *.tex)
   ...
```

- Esiste (ma dovrebbe essere di uso raro) la versione C-style

```
for ((a=1; a <= 10 ; a++))
  ...
```

  Equivalente per shell-puristi: `for i in $(seq 1 10)`

Ciclo sugli argomenti della linea di comando

- Se non si specifica alcuna lista (omettendo anche la keyword `in`) in un ciclo `for` è sottintesa quella degli argomenti passati da linea di comando
- In alcuni casi può essere necessario eseguire un ciclo solo sugli argomenti da una certa posizione in poi (p.e. se nelle prime posizioni c'e' la specifica di un'opzione)
- A tale scopo si può usare il comando `shift` che “elimina” uno o più argomenti nelle posizioni iniziali dalla linea di comando: in questo modo un eventuale ciclo opera solo sugli argomenti rimanenti
- Ovviamente `shift` viene applicato dopo aver opportunamente usato il valore dei primi argomenti (che altrimenti verrebbero persi e ignorati)

"Compito" a casa: Ridenominazione da elenco

- Scrivere uno script `rename.sh` che prende da un file di testo (specificato come argomento da linea di comando) un elenco di coppie di nomi disposte una per riga usando : come separatore. Ad esempio
  ```
  vecchio.txt:nuovo.txt
  prima.tar:dopo.tar
  ...
  ```

- Per ciascuna di queste coppie verifica che il primo nome corrisponda ad un file esistente e scrivibile dall'utente e che il secondo non corrisponda a nessun file o directory esistente

- Se i controlli hanno esito positivo rinomina il file con il primo nome attribuendogli il secondo nome

- Si suggerisce l'utilizzo del comando `cut` (consultare il `man` e/o guardare lucidi prossima lezione)