SVILUPPO DEI FILE COMANDI

la parte di controllo degli argomenti è fondamentale per qualunque programma

È necessario verificare che gli **argomenti siano corretti**

- **Prima nel numero giusto**
- **Poi del tipo richiesto**

CONTROLLO NUMERO DI PARAMETRI

MODO 1: con un case

a)

```
case $# in          #esempio 4 parametri
4)              ;;
*)          echo Errore: ci vogliono 4 argomenti  >&2
            exit 1;;
esac
```

b)

```
case $# in          #esempio 4 parametri
0|1|2|3)        echo Errore: pochi argomenti >&2
                exit 1;;
4)              ;;
*)          echo Errore: troppi argomenti >&2
            exit 1;;
esac
```

MODO 2: con un if

```
if test $# -ne 4          #esempio 4 parametri
then
    echo Errore. Almeno 4 argomenti  >&2
    exit 1
fi
```

Solo in caso di numero di argomenti corretti, andiamo a fare ulteriori controlli

CONTROLLO NOME ASSOLUTO E DIRETTORIO

```
case $1 in          #esempio primo parametro
/*)   if test ! -d $1 -o ! -x $1
      then
            echo $1 non direttorio o non accessibile >&2
            exit 2;
      fi;;
*)
      echo argomento sbagliato: $1 NON assoluto >&2
      exit 3;;
esac
```

CONTROLLO NOME ASSOLUTO E FILE

```
case $2 in          #esempio secondo parametro
/*)   if test ! -f $2
        then     echo $2 non file >&2
                 exit 4
      fi;;
*)    echo argomento sbagliato: $1 NON assoluto >&2
      exit 5;;
esac
```

CONTROLLO NOME RELATIVO SEMPLICE

```
case $3 in          #esempio terzo parametro
*/*)  echo argomento sbagliato: $3 nome NON relativo semplice >&2
      exit 6;;
*) ;;
esac
#in questo caso NON si può controllare né che sia un file
```

#né che sia un direttorio!!!

CONTROLLO NUMERO

MODO 1:

 # il comando expr può verificare una espressione numerica
 # expr restituisce lo stato 0 in caso di successo
 # valore 1 per risultato 0; 2 in caso di insuccesso (e.g. un operando alfanumerico)

```
expr $4 + 0 >/dev/null 2>&1     #esempio quarto parametro
if  test $? -eq 2
then
        echo Errore in argomento numerico: $4  >&2
        exit 7
fi
# qual è la ragione delle ridirezioni su /dev/null?
```

 # MODO 2

 # ogni carattere di una stringa che deve rappresentare
 # un valore numerico deve essere numerico
 # un solo carattere non numerico risulta in un errore

```
case  $4 in
*[!0-9]*)
        echo Errore in argomento numerico: $4 >&2
        exit 7;;
*)      ;;          # tutti i caratteri sono numerici
esac
```

CONTROLLO CARATTERE

```
case $5 in          #esempio quinto parametro
?)    ;;
*)    echo Argomento sbagliato: $5 NON un carattere >&2
      exit 8;;
esac
```

CONTROLLO STRINGA NON NULLA

```
if test -z "$a"          #esempio variabile a
      then
      echo La variabile a risulta nulla >&2
fi
```

Si possono eliminare alcuni argomenti per comodità di scansione: si usi lo **shift**, dopo avere salvato gli argomenti che vengono eliminati

```
salva1=$1  # salvataggio di $1
shift
salva2=$1  # salvataggio di $2
shift
# cosa vale adesso $*?
for i
do
# ciclo fatto per tutti gli argomenti esclusi quelli tolti
done # gli argomenti iniziali sono $salva1 $salva2 $*
```

Per ottenere l'esecuzione di comandi definiti dall'utente:

- uso di **nomi relativi** (ampliamento del PATH con i direttori in cui cercare)
- uso di **nomi assoluti**

Si noti che un **file comandi**

- può **passare** a nuove invocazioni (file comandi o filtri eseguibili) un numero **qualunque di argomenti**
- può **passare** a nuove invocazioni un insieme di valori di **variabili esportate**
  copiate e non condivise successivamente
- può **ottenere** un risultato di ritorno solo usando **$?** se l'invocato ha usato il comando **exit**
- non può vedere le modifica di variabili attuate dall'**invocato** se non registrata in un file del file system