

Esecuzione di uno script

- Diretta:
 - ./scriptname args
 - Il file scriptname deve essere e includere "#!/bin/bash" come prima riga
- Indiretta
 - source ./scriptname args
 - È la shell corrente a eseguire lo script

Assegnazione di variabili

nome_var=valore

Comando echo (per stampa su stdout)

echo [OPZIONI] [STRINGA]

- Opzioni:
 - -n: per non andare a capo.
 - -e: per interpretare i caratteri di escape.

Comando read (per la lettura da stdin)

read

- con una o più variabili passate come argomento
- uso della variabile \$REPLY

Quoting

- Single quoting ''
 - le variabili non vengono espanso
- Double quoting ""
 - le variabili vengono espanso
- Es:
 - a=pippo
 - echo "\$a pippo '\$a'" pluto"
 - pippo pippo \$a pluto
 - echo \$a pippo '\$a' pluto
 - pippo pippo \$a pluto

Utilizzo delle parentesi {} per delimitare il nome di una variabile

- Es.
 - nome=Gian
 - echo \${nome}marco
 - Gianmarco

Cattura del stdout di un comando

- \$(<comando>)

Comando exit

- exit [numero]
 - termina l'esecuzione di un processo restituendo un valore al processo chiamante

tail: output the last part of files

tail [<options>] <file>
[-n, --lines
[-c, --bytes]]

- Es:
 - o exit 0
 - o restituisce un valore vero

Esecuzione di calcoli aritmetici

- un metodo a scelta dello studente
- Es.
 - o let s=\$n1+\$n2
 - o Assegna alla variabile \$s la somma di \$n1 e di \$n2

Variabili di shell speciali

- \$0, \$1, \$2, ...
 - o passaggio di parametri sulla riga di comando
- \$*
 - o lista completa dei parametri, escluso il nome dello script
- \$#
 - o Numero di parametri
- \$\$
 - o PID del processo
- \$?
 - o valore di ritorno dell'ultimo processo eseguito

Costrutto if-then-else (e elsif)

```
if condizione ; then
  statements
elif condizione
then
  statements
else
  statements
fi
```

Costrutto while (compresa ridirezione di stdin e stdout)

```
while condizione
do
  statements
done << $fileIn >> $fileOut
```

Formati richiesti nella condizione dei costrutti if e while

Solo le condizioni espresse tra simboli [...] sono richieste
 (non sono invece richieste le condizioni basate sulla parola chiave test)

- Confronti numerici:
 - o -eq uguaglianza (==)
 - o -ne non uguaglianza (!=)
 - o -gt maggiore (>)
 - o -ge maggiore o uguale (>=)
 - o -lt minore (<)
 - o -le minore o uguale (>)
- Confronti tra stringhe:

- = uguaglianza
- != non uguaglianza

- Condizioni su file:

- -d <arg> vero se <arg> è un directory
- -f <arg> vero se <arg> è un file
- -r <arg> vero se <arg> ha il permesso di lettura
- -w <arg> vero se <arg> ha il permesso di scrittura
- -x <arg> vero se <arg> ha il permesso di esecuzione

- Operatori logici utilizzabili all'interno di una condizione:

- ! not
- -a and
- -o or

- Operatori logici utilizzabili in un elenco di condizioni:

- && and
- || or

Costrutto for

```
for var in [ list ]
do
    statements
done
```

Istruzioni

- break
- continue

Vettori

```
# Dichiarazioni
array[3]="valore"
array=( 4 8 7 )
array=( [0]=4 [1]=8 [2]=7 [5]=10 )
# Accesso
echo ${array[1]} # Accesso all'elemento 1 dell'array (valore 8)
echo ${array[*]} # Stampa tutti gli elementi dell'array
echo ${#array[*]} # Numero di elementi contenuti nell'array
```

while read row
do

 ::
 ::

done < file.txt > out.txt

n°chars = "\${#row}"

- o Options:
 - -e <pattern>: specify a pattern to be matched.
 - -E <pattern>: specify extended regular expression pattern to be matched.
 - -H: print the file name for each match.
 - -n: print the line number for each match.
 - -i: ignore case.
 - -v: invert the sense of matching, to select non-matching lines.
 - -l, --quiet, --silent: no output is produced.
 - -l, --files-with-matches: print out only file names.
- wc [<opts>] <filepath>
 - o Print newline, word, and byte counts for a file.
 - o Options:
 - -w, --words: print words counts.
 - -c, --bytes: print the byte counts.
 - -m, --chars: print the character counts.
 - -l, --lines: print the newline counts.

File Search

- find [<directory>] [<options>] [<actions>]
 - o Search for files in a directory hierarchy (with a specified root, i.e., <directory>).
 - o Options:
 - -name <pattern>: search files whose name matches the pattern.
 - -regex <pattern>: search files whose path matches a regular expression.
 - -regextype posix-extended: specify posix-extended format for regular expressions
 - -type <f|l|d>: search files of a specific type.
 - -mindepth <depth>: search files starting from the specified directory tree depth.
 - -maxdepth <depth>: search files up to the specified directory tree depth.
 - -size <[+,-]n[cwkMG]>: search files whose size starts from (+) or goes up to (-) the specified size. (c=bytes, w=words, k=kilobytes, M=megabytes, G=gigabytes).
 - -user uname: File is owned by user uname (numeric user ID allowed).
 - -group gname: File belongs to group gname (numeric group ID allowed).
 - o Actions:
 - -exec <cmd>: execute command on each matched file.
 - \{} (or '{}') can be used as a placeholder for the file path.
 - The command must end with \; (or ';').

File Permissions Management

- chmod [<opts>] <mode> <file>

- o Change file permissions. <mode> can be specified symbolically ([ugoa][+-][rwx]) or numerically (octal digits).
- o Options:
 - -R: change permissions of files and directories recursively.

String manipulation

- basename path
 - o Strip directory and suffix from path.
- dirname path
 - o Strip last component from path.

Redirections

- cmd1 | cmd2
 - o Redirect standard output of cmd1 to standard input of cmd2.
- cmd < file
 - o Redirect standard input of cmd from file.
- cmd > file
 - o Redirect standard output of cmd to file.
- cmd 2> file
 - o Redirect standard error of cmd to file.
- cmd &> file
 - o Redirect standard output and standard error of cmd to file.
- cmd >> file
 - o Append standard output of cmd to file.

Shortcuts

- CTRL+C
 - o Terminate the current foreground process.
- CTRL+Z
 - o Stop the current foreground process.
- TAB
 - o Autocompletion.

grep -e "regexp"
grep -A n | show n lines after grep match