Basic Unix Commands

Bash Commands
- man <cmd>
  - Shows manual for a command.
- sudo <cmd>
  - Execute a command as the superuser.

File System Navigation
- cd <path>
  - Change current working directory.
- ls [<opts>] [<path>]
  - List the contents of a directory (current working directory by default).
  - Options:
    - -l: use a more detailed listing format.
    - -a: show hidden files and folders.
- pwd
  - Print name of the current working directory.

File System Manipulation
- mkdir <dirpath>
  - Create a new directory, if it does not already exist..
- rmdir <dirpath>
  - Remove a directory, if it is empty.
- mv <srcpath> <destpath>
  - Move (rename) the file <srcpath> to <destpath>.
- rm [<opts>] <path>
  - Remove a file.
  - Options:
    - -R: remove directories and their contents recursively.
- cp [<opts>] <srcpath> <destpath>
  - Copy the file <srcpath> to <destpath>.
  - Options:
    - -R: copy directories and their contents recursively.
- ln [<opts>] <target> <destpath>
  - Create an hard link to file <target> in <destpath>.
  - Options:
    - -s: make symbolic links instead of hard links.

Process Management
- ps [<opts>]
  - List active processes.
  - Options:
    - -e: list all processes.
    - -l: use a more detailed listing format.
- top
  - Show a a dynamic real-time view of running processes.
- kill -<signum> <pid>
  - Send the signal <signum> to the process specified by <pid>.
  - Signals:
    - -9: terminate a process.
- <cmd> &
  - Execute command in background.

File Operations
- cat <filepath>
  - Print file on the standard output.
- more <filepath> / less <filepath>
  - Print file on the standard output page by page.
- sort [<opts>] <filepath>
  - Sort lines of text files (alphabetically by default).
  - Options:
    - -r: reverse order.
    - -n: use numerical order.
    - -k <key>: sort via key.
- cut [<opts>] <filepath>
  - Remove sections from each line of a text file.
  - Options:
    - -f <field1,field2,...fieldN>: specify fields to select.
    - -d <delim>: use <delim> instead of TAB for field delimiter.
- tr [<opts>] <set1> [<set2>]
  - Translate standard input characters from <set1> to <set2>.
  - Options:
    - -d: delete characters from <set1> instead of translating them.
- uniq [<opts>] <filepath>
  - Omit repeated lines.
  - Options:
    - -d: report only repeated lines instead of omitting them.
- grep [<opts>] <filepath>
  - Print lines of a file matching a pattern.
  - 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.
    - --quiet, --silent, -l no output is produced.
    - --files-with-matches, -l print out only file names.
- wc [<opts>] <filepath>
  - Print newline, word, and byte counts for a file.
  - Options:
    - -l: print newline count only.
    - -w: print word count only.
    - -c: print byte count only.

File Search
- find [<opts>] [<rootpath>]
  - Search for files in a directory hierarchy (with a specified root).
  - 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).
    - -exec <cmd>: execute command on each matched file.
      - "\{}" can be used as a placehold er for the file path.
      - The command must end with "\;".

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

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

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

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

The Bash Language

Variables assignment
name_var=value

echo command (print on stdout)
echo [OPTIONS] [STRING]
- Options:
  - -n: no new line.
  - -e: change the meaning of escape chars.

Read command
read
  - Read a line from standard input

Quoting
- Single quoting ' '
  - no variable expansion
- Double quoting ""

- variable expansion
- Es:
  - a=pippo
  - echo "$a pippo "'$a'" pluto"
  - pippo pippo $a pluto
  - echo $a pippo '$a' pluto
  - pippo pippo $a pluto

Parenthesis { } for variable expansion
- Es.
  - name=Gian
  - echo ${name}marco
  - Gianmarco

command expansion
- $(<command>)

Script execution
- #!/bin/bash
  - from file
- (cd /home; ls)
  - directly using the parenthesis ( )

exit command
- exit [number]
  - current process termination, return a value to the process
- Ex:
  - exit 0
  - return true value

Arithmetic expansion
- choose among the methods:
- Ex.
  - let s=$n1+$n2
  - assigns to $s → $n1 + $n2

Special shell variables
- $0, $1, $2, ...
  - command line parameters
- $*
  - whole parameters list , but the script name
- $#
  - Parameters number
- $$
  - process PID
- $?
  - return value of the last command

statement if-then-else (and elsif)
```
if cond ; then
  statements
elif cond
then
  statements
else
  statements
fi
```

while (including stdin e stdout redirections)
```
while cond
do
  statements
done << $fileIn >> $fileOut
```

Conditions types for if and while

- -eq uqual (==)
- -ne nt equal (!=)
- -gt greater (>)
- -ge greather equal (>=)
- -lt lower than (<)
- -le lower equal (>)

- String comparison:
  - = equal
  - != not equal

- File conditionals:
  - -d <arg> true if <arg> is a directory
  - -f <arg> true <arg> is a file
  - -r <arg> true if <arg> has read permission
  - -w <arg> true if <arg> has write permission
  - -x <arg> true if <arg> has execution permission

- Logical operator:
  - ! not
  - -a and
  - -o or

- Logical operator for condition list:
  - && and
  - || or

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

Istructions
- break
- continue

Array
```
# Declarations
array[3]="value"
array=( 4 8 7 )
array=( [0]=4 [1]=8 [2]=7 [5]=10 )
# Use
echo ${array[1]}  # acces to the 1st array item (value=8)
echo ${array[*]}  # print the array items
echo ${#array[*]} # array size
```

AWK

AWK execution

awk [options] {script} [file1] ... [filen]

[options]: -v var=val
- assign val to the new variable var

{script}:
- awk [options] 'command' [file1] ... [filen]
- awk [options] –f scriptFile [file1] ... [filen]

[filei]: input files

AWK SCRIPT

built-in vars

RS = record separator [default \n]

FS = field separator

NR = number of records

NF = number of fields

$0 = whole record

S1= first field ; $2=second field .....

FILENAME = current file name

ARGV[0] script name
ARGV[1] ... ARGV[ARGC-1] parameters

Conditions

(null)
- true for all records

/regExp/
- true if the regExp match the record

expression
- true if !=0 or !=null if return a string

pat1, pat2
- record interval from pat1 to pat2

BEGIN
- Action performed before the file(s) processing

END
- Action performed after the file(s) processing

Useful Functions / Comands

Print

print [p1] ... [pn] [> file]

printf (format, ...) [> file]
- Stampa (formattata) su file

stile C

getline [var] [<newFile]
reads lines from a new file

int(x), sqrt(x), exp(x), log(x), sin(x), rand(), etc.
arithmetic functions

length (str)
returns the length of the string

toupper (str)
str to upper case

tolower (str)
str to lower case

system (command)
bash command execution

match (str, regExp)
true if str match regExp

gsub (regExp, str [, src])
global substitution replace regExp with str

split (str, vet [, del])
split str in vet according to the delimeter (del)

substr (str, i [, n])
returns an str substring starting from index I having size n

Operators and Expressions

Arithmetics
+, -, *, /, %, ^ (or **)

String concatenation
str1 str2 str3 ...

Relational
==, !=, <, >, <=, >=

Logical
&&, ||, ! (negazione)

Regexp comparison (match)
~, !~