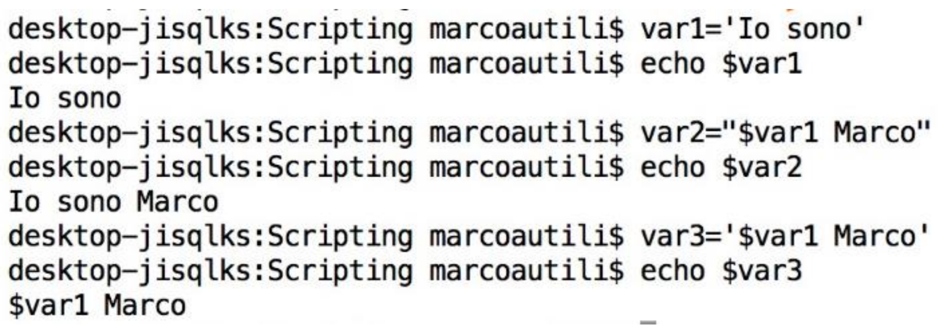
Variables and command line arguments

- $0 - The name of the script
- $1 - $9 - Any command line arguments given to the script: $1 is the first argument, $2 the second and so on
- $# - How many command line arguments were given to the script.
- $* or $@ - All of the command line arguments passed to the script
- $? - The exit status of the most recently run process
- $$ - The process ID of the current script
- $USER - The username of the user running the script
- $HOSTNAME - The hostname of the machine the script is running on
- $SECONDS - The number of seconds since the script was started
- $RANDOM - Returns a different random number each time is it referred to
- $LINENO - Returns the current line number in the Bash script

Single quotes VS double quotes

- ‘Single quotes’ will treat every character literally
- “Double quotes” will allow you to do substitution



```
desktop-jisqlks:Scripting marcoautili$ var1='Io sono'
desktop-jisqlks:Scripting marcoautili$ echo $var1
Io sono
desktop-jisqlks:Scripting marcoautili$ var2="$var1 Marco"
desktop-jisqlks:Scripting marcoautili$ echo $var2
Io sono Marco
desktop-jisqlks:Scripting marcoautili$ var3='$var1 Marco'
desktop-jisqlks:Scripting marcoautili$ echo $var3
$var1 Marco
```

Variables declaration

You can declare a variable by using **declare** or **typeset** command.
Most used options are:

- -r : set the variable to only read variable
- -i : declare an integer variable
- -a : declare an array variable

Syntax:
declare [options] [variable_name[=value]]
typeset [options] variable_name[=value]]