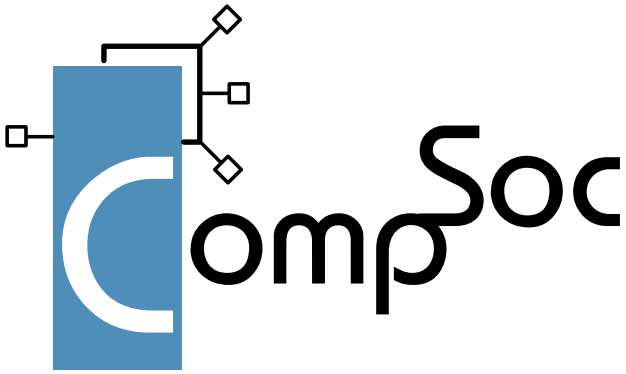
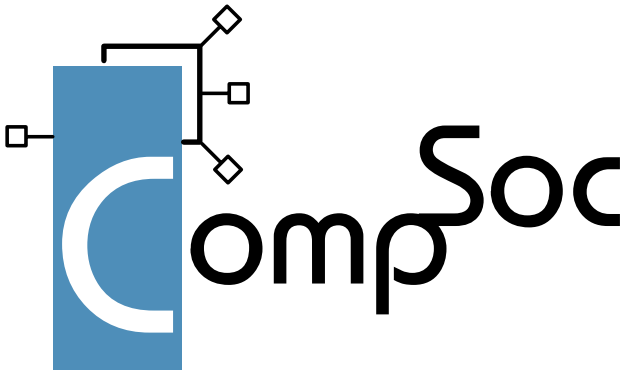


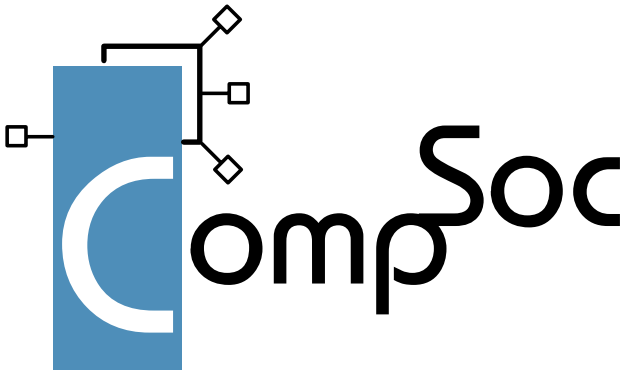
Shell Scripting Workshop

What you can do with the shell that you can't do with Windows



Contents

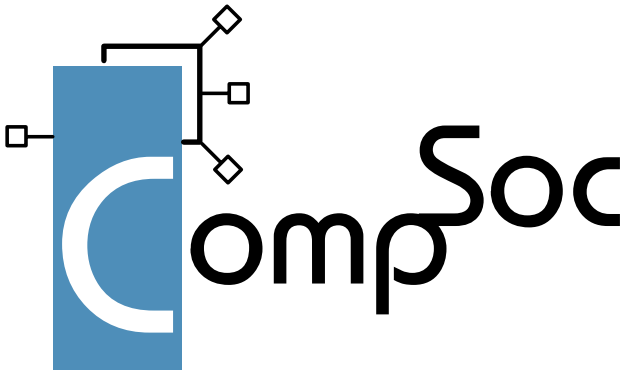
- Simple text processing utilities
  - cat, echo
  - grep, awk, sed
  - sort, seq
- How to put them together
  - Pipes
  - For loops
  - Conditionals
  - While loops
- Useful Examples
- How to add scripts to your $PATH



Simple text processing utilities (1)

- cat
  - prints out a file to standard output

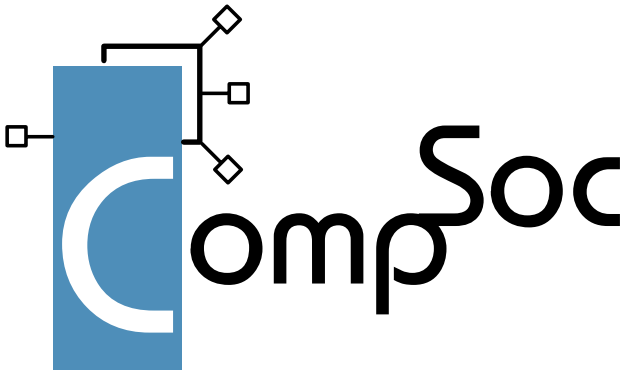
```
(cyan@frink:~)$ cat /tmp/shell-tutorial-1
This is the first file for the shell scripting tutorial.
(cyan@frink:~)$
```



Simple text processing utilities (2)

- echo
  - prints its arguments to standard output

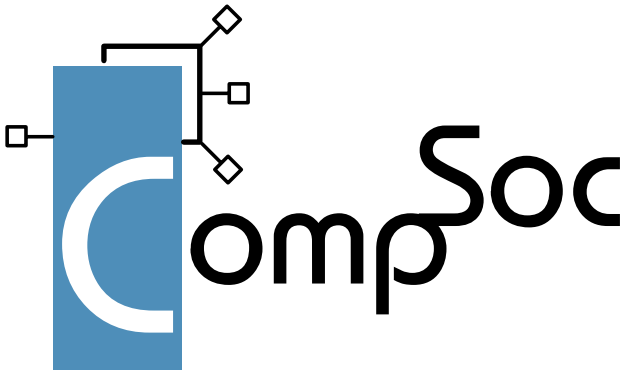
```
(cyan@frink:~)$ echo this is an example.
this is an example.
(cyan@frink:~)$
```



Simple text processing utilities (3)

- grep
  - searches for a string

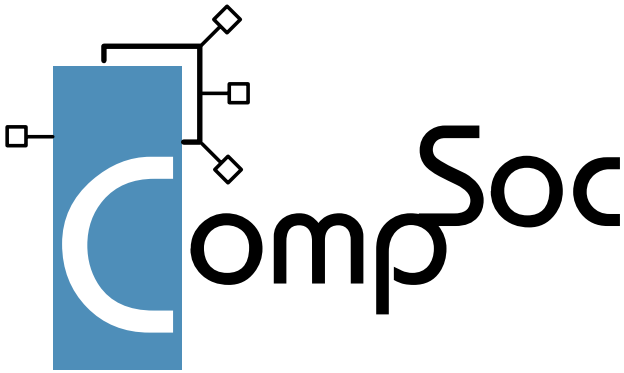
```
(cyan@frink:~)$ grep 'cyan' /etc/passwd
cyan:x:1315:100:Patrick Farrell,,,:/home/users/cyan:/bin/bash
(cyan@frink:~)$
```



Simple text processing utilities (4)

- awk
  - used mainly for printing out one of its arguments

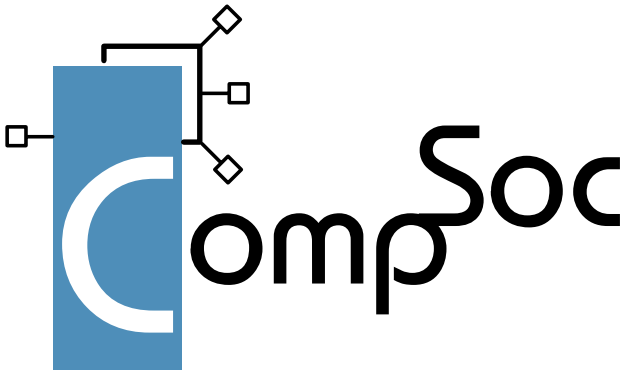
```
(cyan@frink:~)$ echo arg1 arg2 arg3 | awk '{print $2}'
arg2
(cyan@frink:~)$
```



Simple text processing utilities (5)

- sed
  - used mainly for replacing text with something else

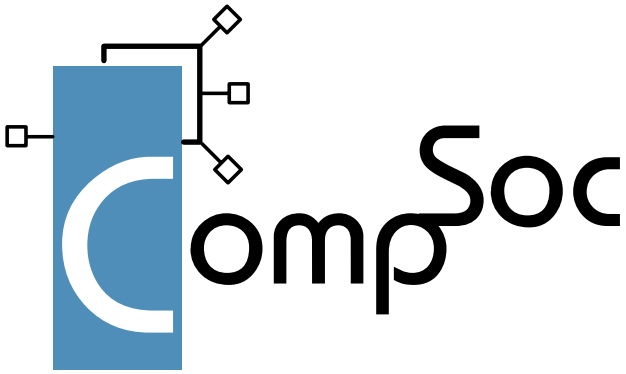
```
(cyan@frink:~)$ echo arg1 arg2 arg3 | sed 's/arg2/arg4/'
arg1 arg4 arg3
(cyan@frink:~)$
```



Simple text processing utilities (6)

- sort
  - used for sorting text alphabetically

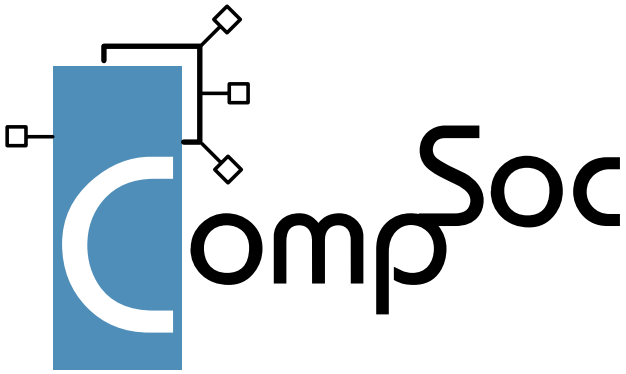
```
(cyan@frink:~)$ cat /tmp/shell-tutorial-2
a
c
b
(cyan@frink:~)$ cat /tmp/shell-tutorial-2 | sort
a
b
c
(cyan@frink:~)$
```



Simple text processing utilities (7)

- seq
  - just prints out a range of numbers

```
(cyan@frink:~)$ seq 1 4
1
2
3
4
(cyan@frink:~)$
```

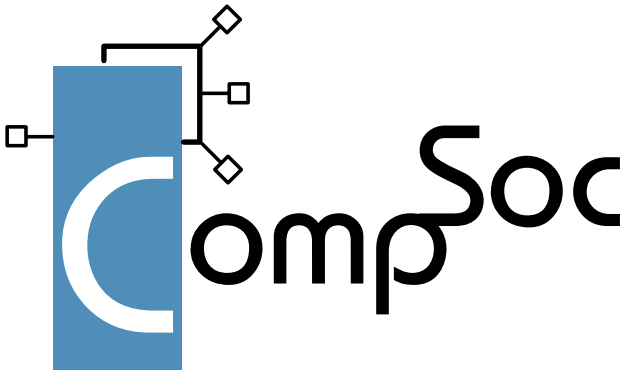


Pipes

- A pipe takes the output of one program and makes it the input of another

```
(cyan@frink:~)$ cat /etc/passwd | grep cyan
cyan:x:1315:100:Patrick Farrell,,,:/home/users/cyan:/bin/bash
```

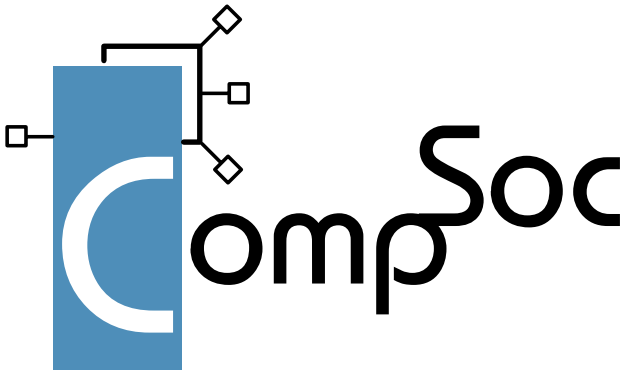
- The output of the cat command is passed to grep.



For loops

- Best shown by example:

```
(cyan@frink:~)$ for number in 1 2 3; do echo "- number is now
$number"; done
- number is now 1
- number is now 2
- number is now 3
(cyan@frink:~)$
```

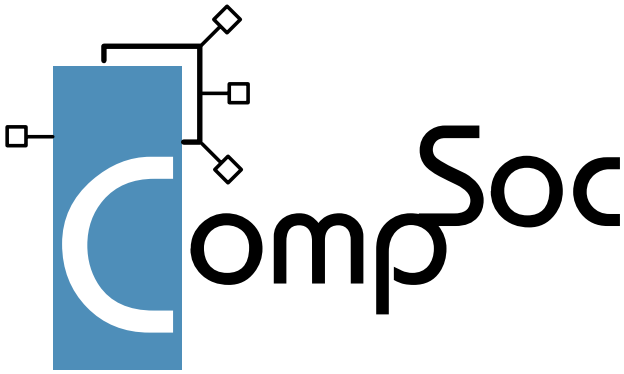


Conditionals

- Again, best shown by example:

```
(cyan@frink:~)$ if [ -f /etc/passwd ]; then echo "/etc/passwd is a file."; fi
/etc/passwd is a file.
(cyan@frink:~)$
```

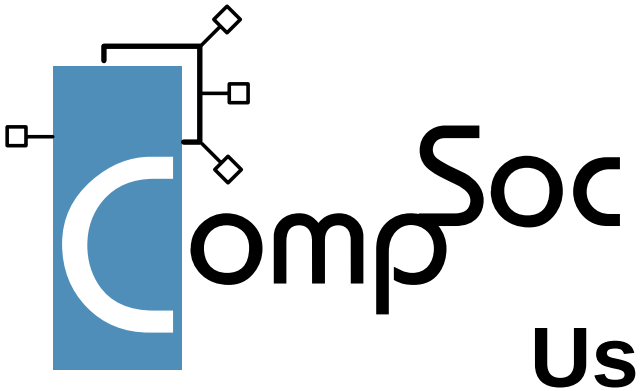
- Too many of them to list; you'll pick them up as you go along
  - see "man test" for a full list



While loops

- Keeps on looping while a condition is true

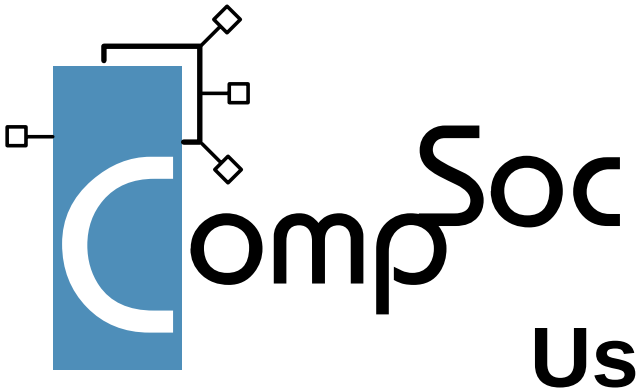
```
(cyan@frink:~)$ NUM=1;
while [ $NUM -lt 3 ];
do echo $NUM; let NUM=$NUM+1; done
1
2
(cyan@frink:~)$
```



Useful examples (1)

- Rename all files in a directory to lowercase:

```
#!/bin/bash
for file in *; do
        lname=`echo $file | tr A-Z a-z`
        if [ “$file” != “$lname” ]; then
                mv $file $lname
        fi
done
```



Useful examples (2)

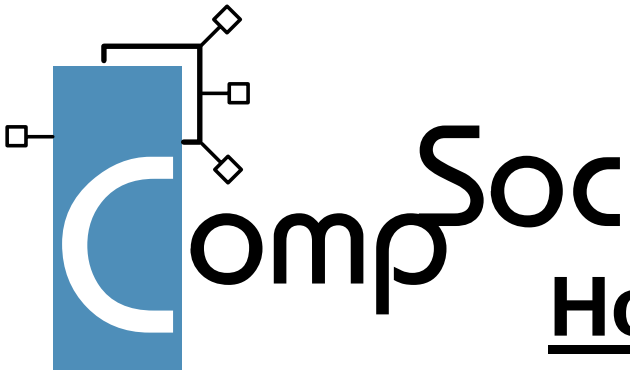
- Download all of a lecturer's notes:

  (cyan@frink:~)$ for i in `seq 1 9`; do
  wget http://example.nuigalway.ie/maxxx/lecture-$i.pdf; done

- Convert all images to thumbnail

  (cyan@frink:~)$ for picture in `/bin/ls *.jpg`; do
  convert -resize 25%x25% $picture thumb.$picture; done

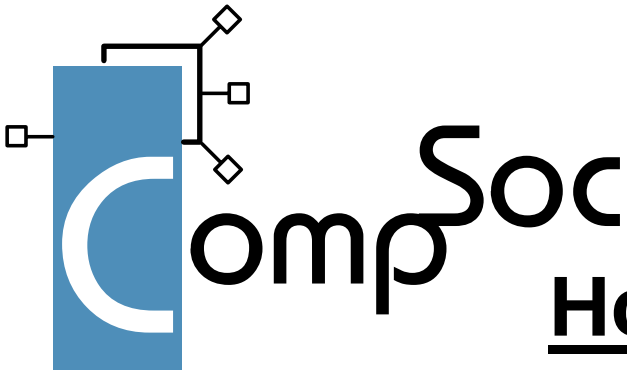
  See also /usr/local/share/scripts for some more simple examples.



How to add scripts to your $PATH (1)

- If you want to be able to execute the script from anywhere, you need to use a shell environment variable called $PATH
- When you type the name of a command, bash looks through the locations listed in your $PATH variable (seperated by ':') to find it

```
(cyan@frink:~)$ echo $PATH
/usr/local/bin:/usr/bin:/bin:/usr/bin/X11:/usr/games:
```

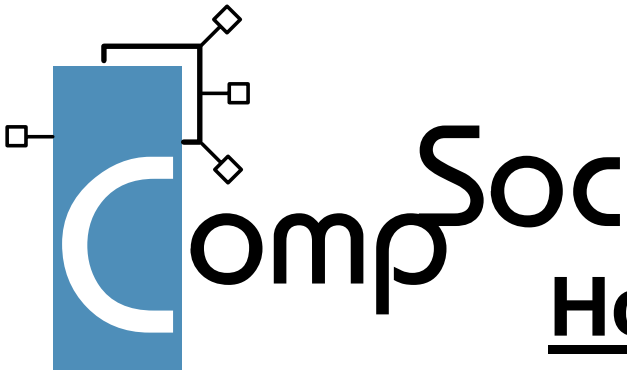


How to add scripts to your $PATH (2)

- Try:

```
(cyan@frink:~)$ mkdir ~/bin
(cyan@frink:~)$ mv script ~/bin
(cyan@frink:~)$ chmod u+x ~/bin/*
```

  - **Reminder:** The '~' character is a shortcut syntax for your home directory (e.g. /home/users/cyan).

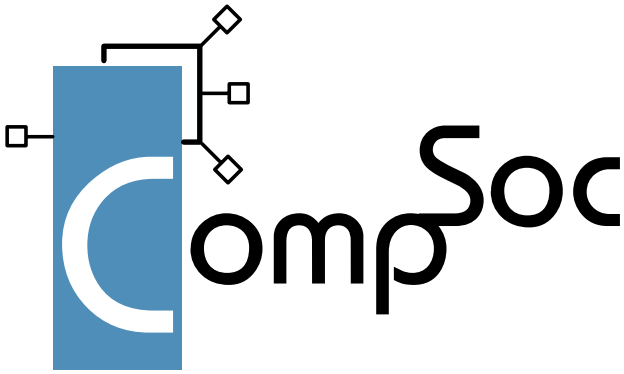


How to add scripts to your $PATH (3)

- To add the newly created ~/bin directory to your $PATH
  - edit your .bash_profile and uncomment the following lines by removing the '#':

```
#if [ -d ~/bin ] ; then
#    PATH="~/bin:${PATH}"
#fi
```

- Then when you log in again, you will be able to execute the script from anywhere.



Links

- BASH Programming - Introduction HOW-TO
  - Start programming basic-intermediate shell scripts
  - http://tldp.org/HOWTO/Bash-Prog-Intro-HOWTO.html
- Bash Prompt HOWTO
  - Creating and controlling terminal and xterm prompts
  - http://tldp.org/HOWTO/Bash-Prompt-HOWTO/index.html
- Advanced Bash-Scripting Guide
  - The definite guide and reference
  - http://tldp.org/LDP/abs/html/index.html