UNIX command line

What is Unix/Linux?

What operating system(s) do you know?

What is the computer shell?

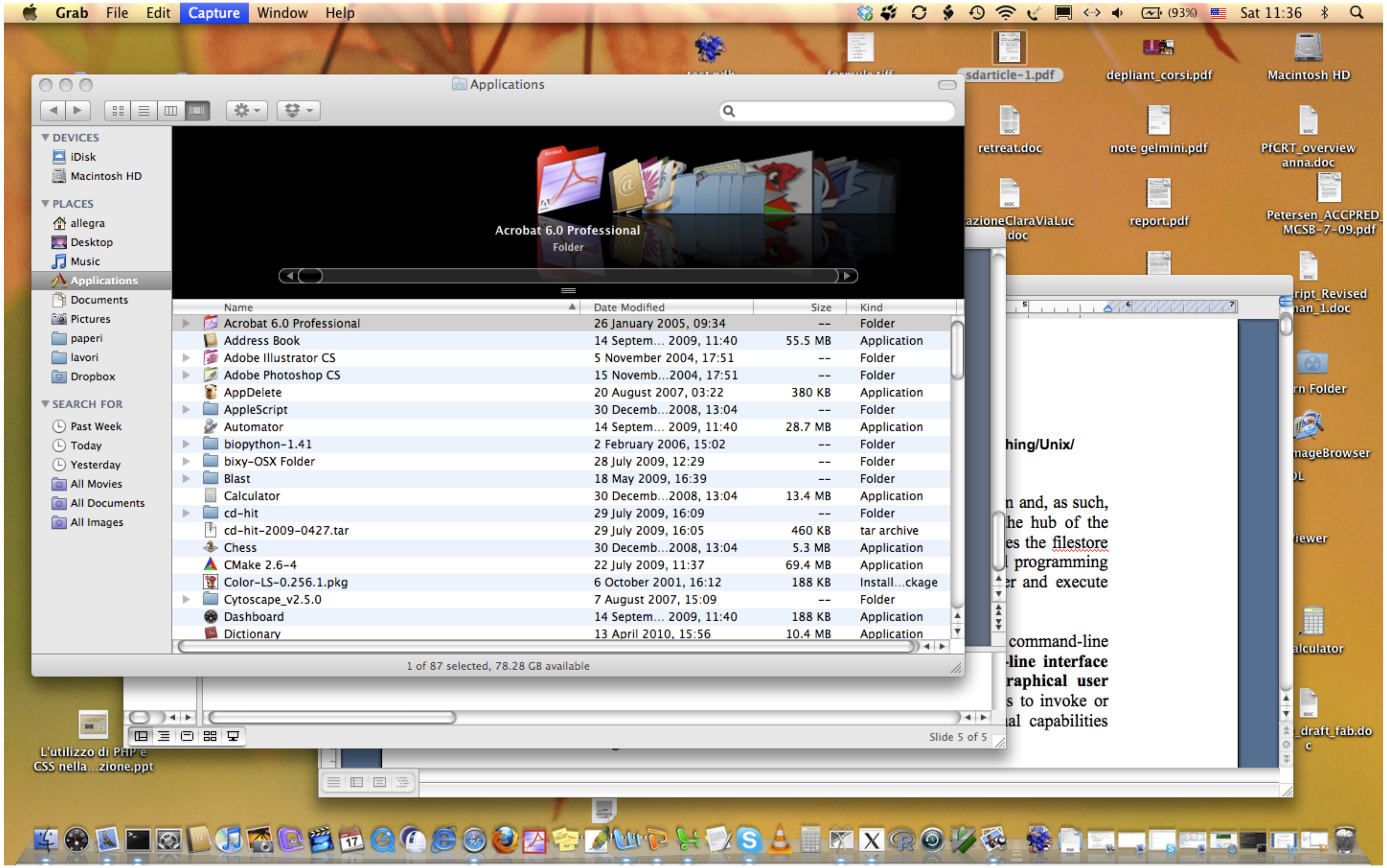
The shell

The shell is an interpreter (a program) that lets you interact with the operating system

$$\begin{tikzcd}
\boxed{\text{graphical shell}} \arrow[d] & & \boxed{\text{command-line shell}} \arrow[d] \\
\boxed{\text{graphical interface}} & & \boxed{\text{command-line interface}}
\end{tikzcd}$$

What is the graphical interface?

graphical interface

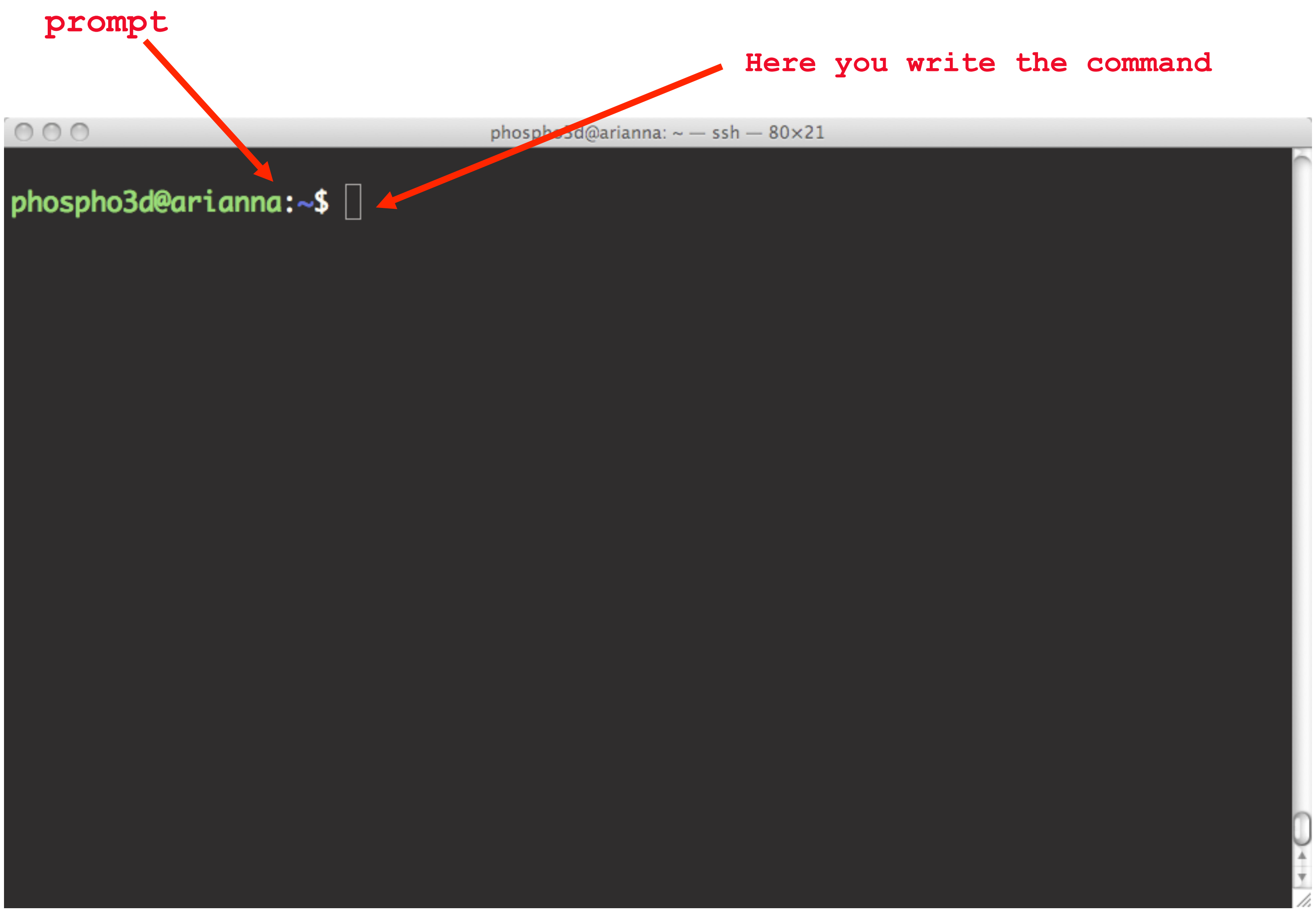


Name	Date Modified	Size	Kind
Acrobat 6.0 Professional	26 January 2005, 09:34	--	Folder
Address Book	14 Septem... 2009, 11:40	55.5 MB	Application
Adobe Illustrator CS	5 November 2004, 17:51	--	Folder
Adobe Photoshop CS	15 Novemb...2004, 17:51	--	Folder
AppDelete	20 August 2007, 03:22	380 KB	Application
AppleScript	30 Decemb...2008, 13:04	--	Folder
Automator	14 Septem... 2009, 11:40	28.7 MB	Application
biopython-1.41	2 February 2006, 15:02	--	Folder
bixy-OSX Folder	28 July 2009, 12:29	--	Folder
Blast	18 May 2009, 16:39	--	Folder
Calculator	30 Decemb...2008, 13:04	13.4 MB	Application
cd-hit	29 July 2009, 16:09	--	Folder
cd-hit-2009-0427.tar	29 July 2009, 16:05	460 KB	tar archive
Chess	30 Decemb...2008, 13:04	5.3 MB	Application
CMake 2.6-4	22 July 2009, 11:37	69.4 MB	Application
Color-LS-0.256.1.pkg	6 October 2001, 16:12	188 KB	Install...ckage
Cytoscape_v2.5.0	7 August 2007, 15:09	--	Folder
Dashboard	14 Septem... 2009, 11:40	188 KB	Application
Dictionary	13 April 2010, 15:56	10.4 MB	Application

1 of 87 selected, 78.28 GB available

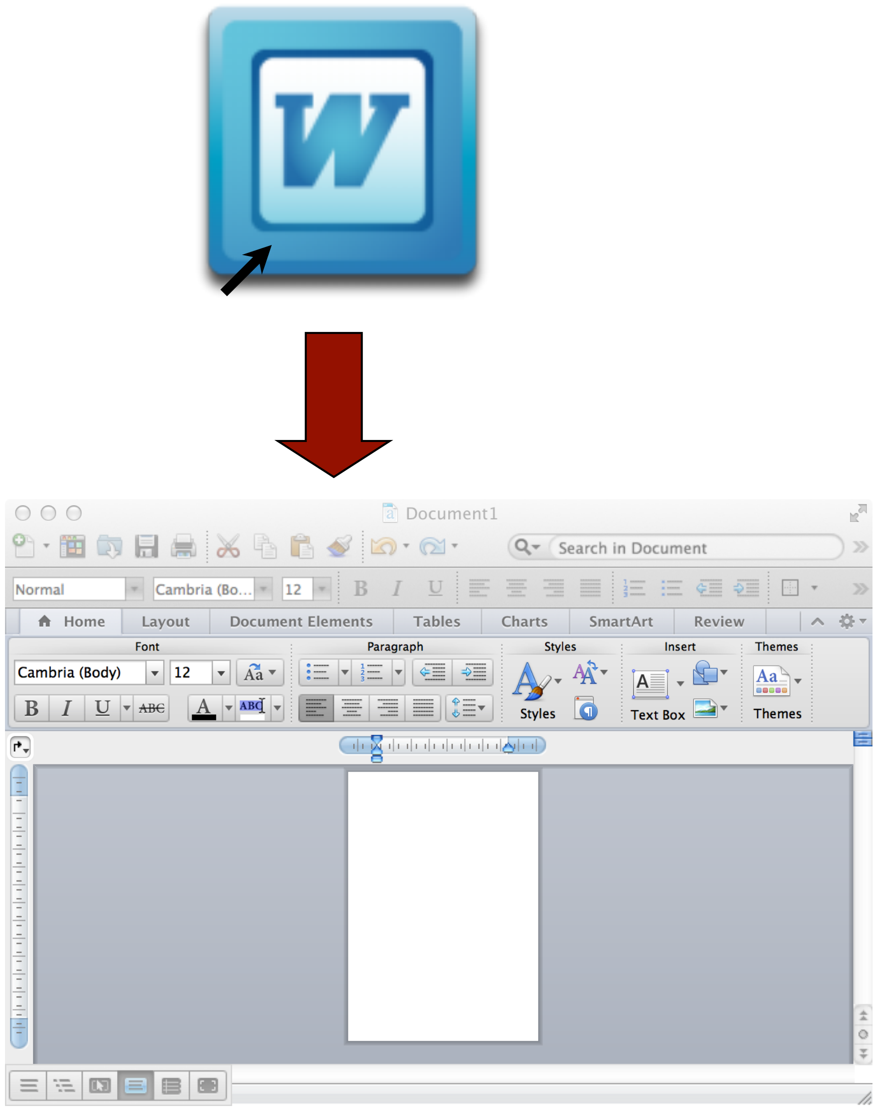
What is the command line interface (or Terminal)?

command-line interface

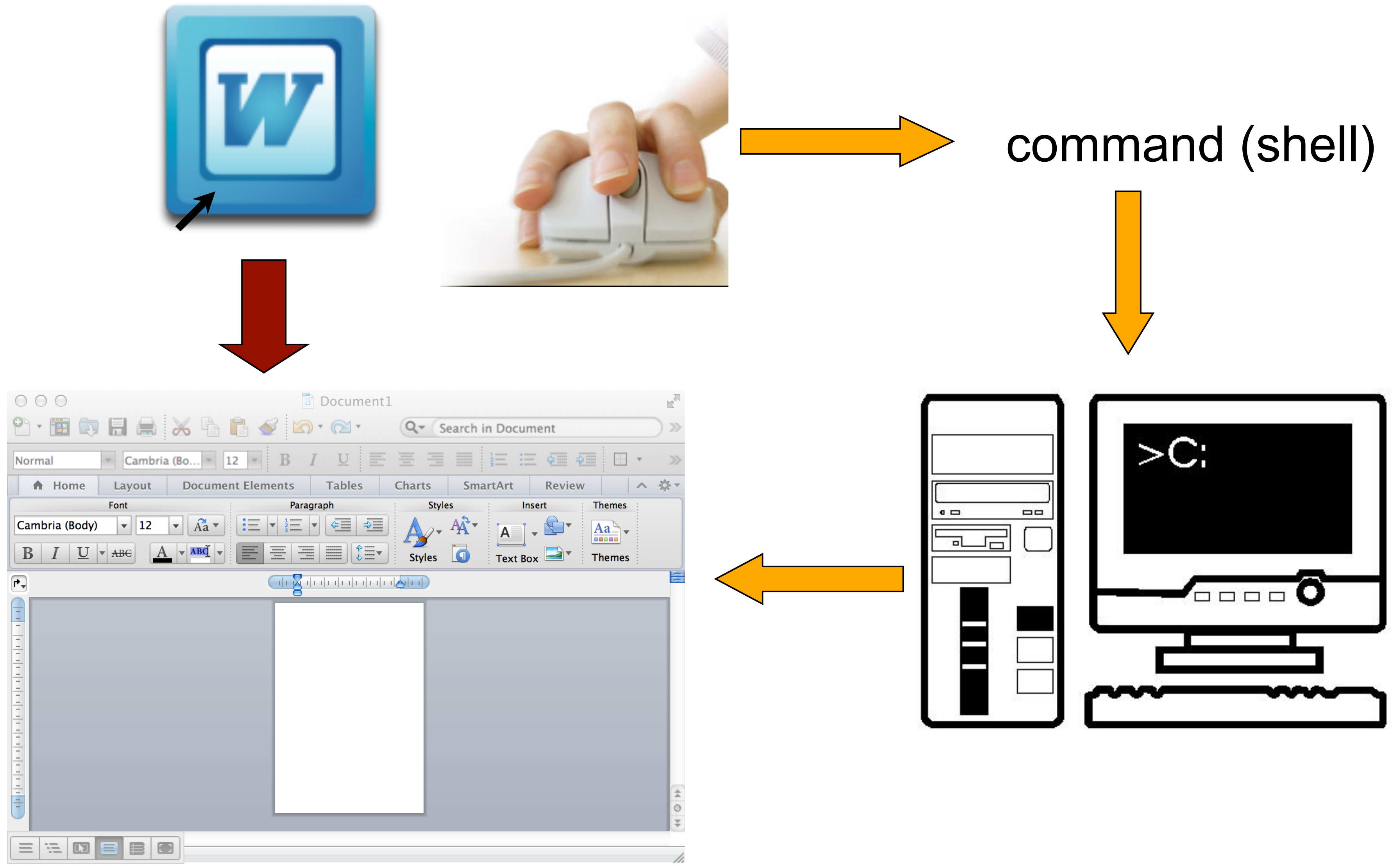

prompt
Here you write the command
phospho3d@arianna: ~ — ssh — 80×21

```
phospho3d@arianna:~$ 
```

What happens when you double click on the icon of an application?

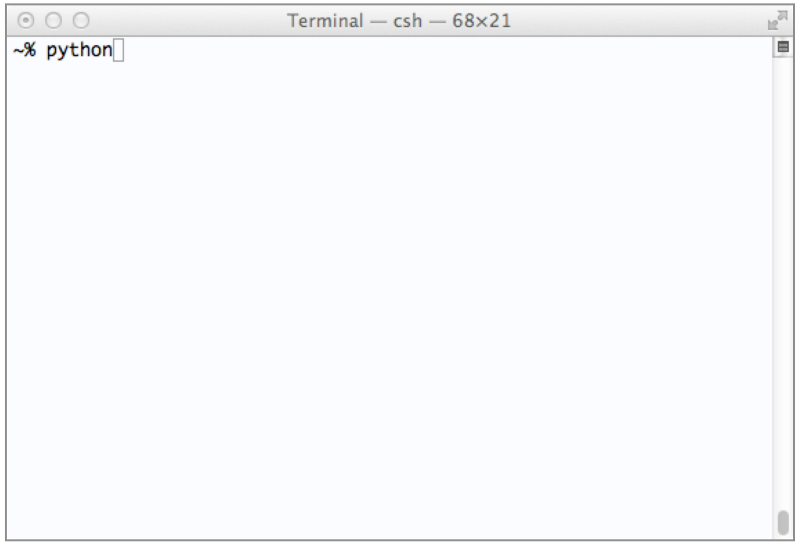


What happens when you double click on the icon of an application?

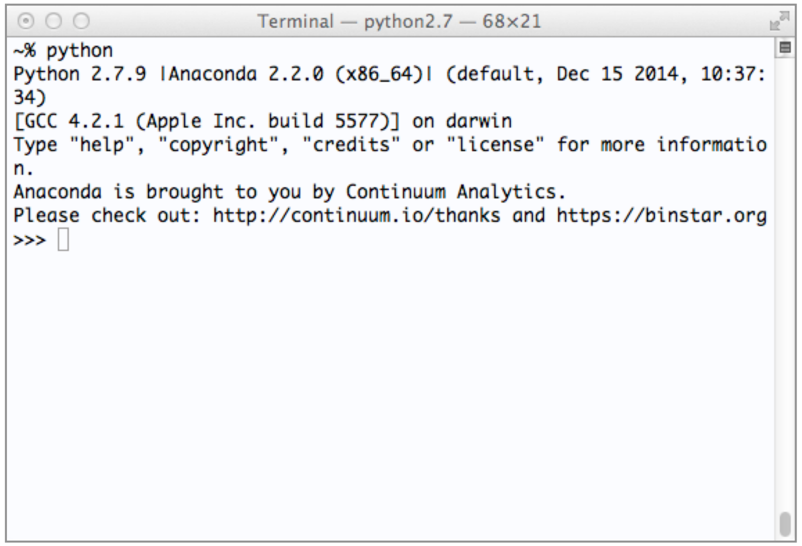


Open a command-line terminal on your computer

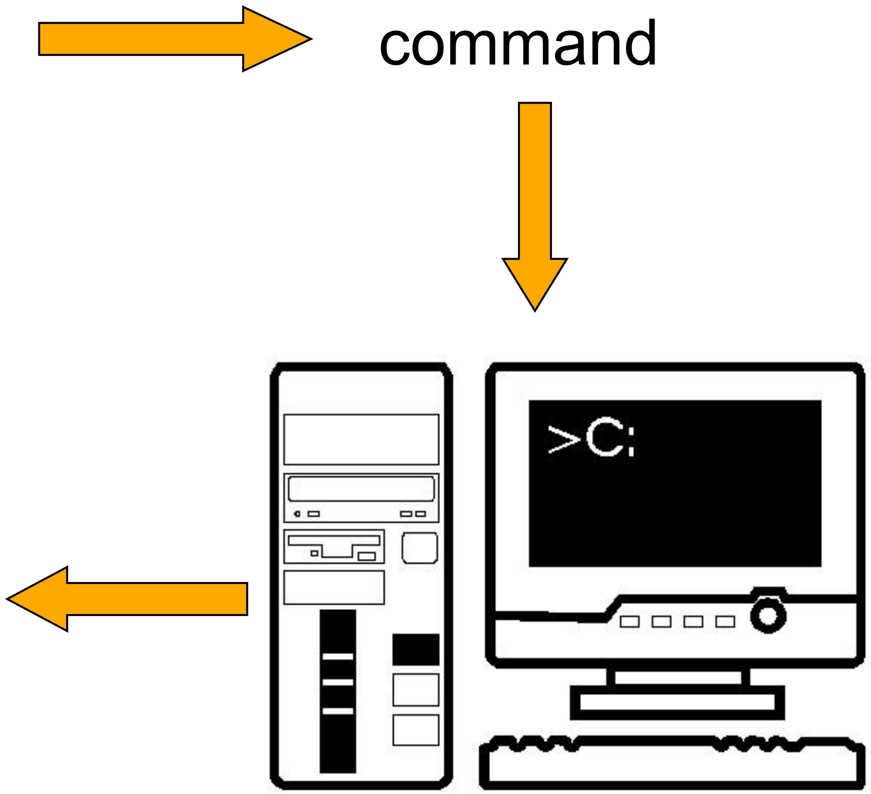
You can type a program name at the terminal prompt and then type **`[Return]`**

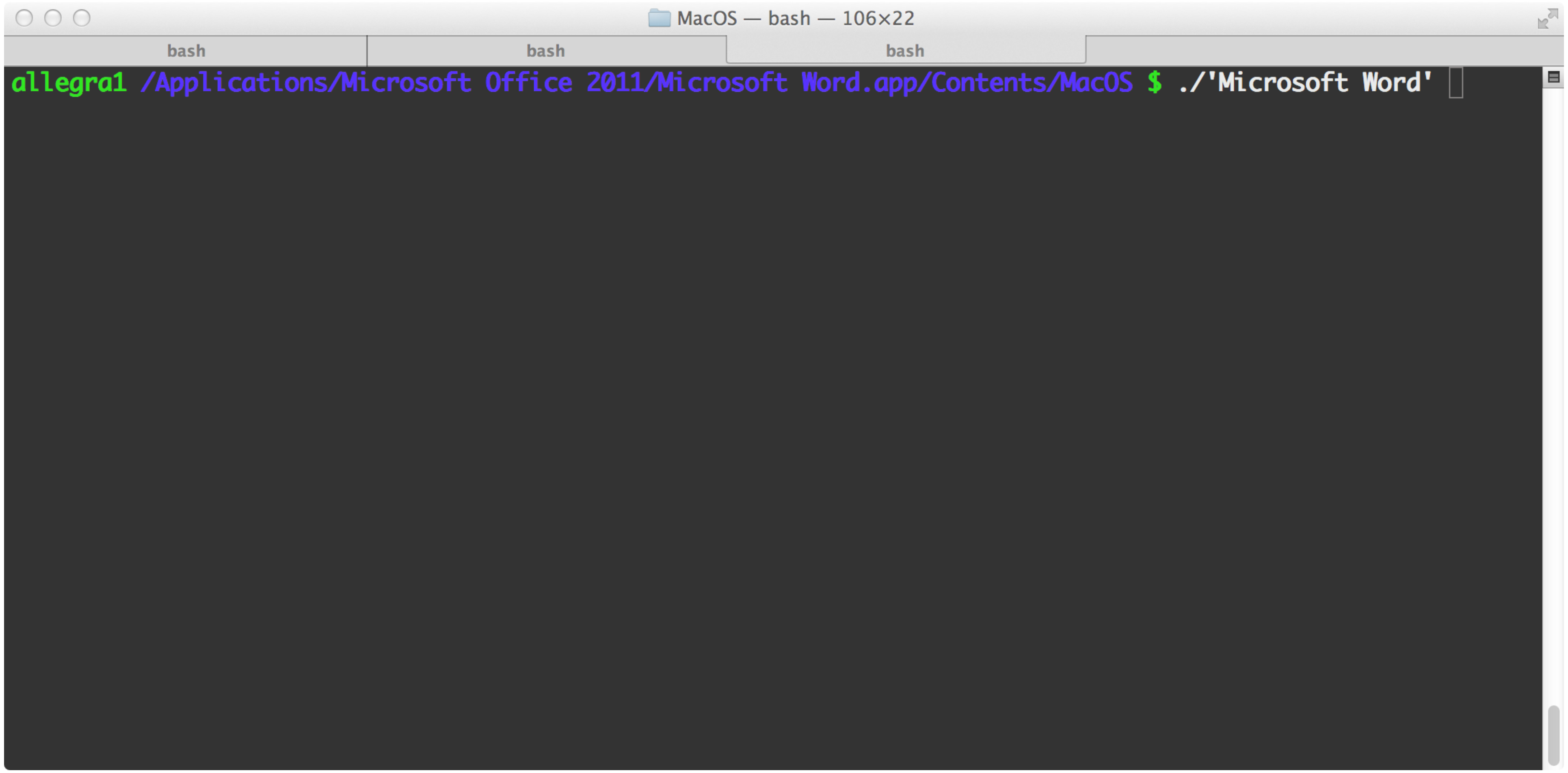


```
~% python
```



```
~% python
Python 2.7.9 |Anaconda 2.2.0 (x86_64)| (default, Dec 15 2014, 10:37:
34)
[GCC 4.2.1 (Apple Inc. build 5577)] on darwin
Type "help", "copyright", "credits" or "license" for more informatio
n.
Anaconda is brought to you by Continuum Analytics.
Please check out: http://continuum.io/thanks and https://binstar.org
>>> 
```





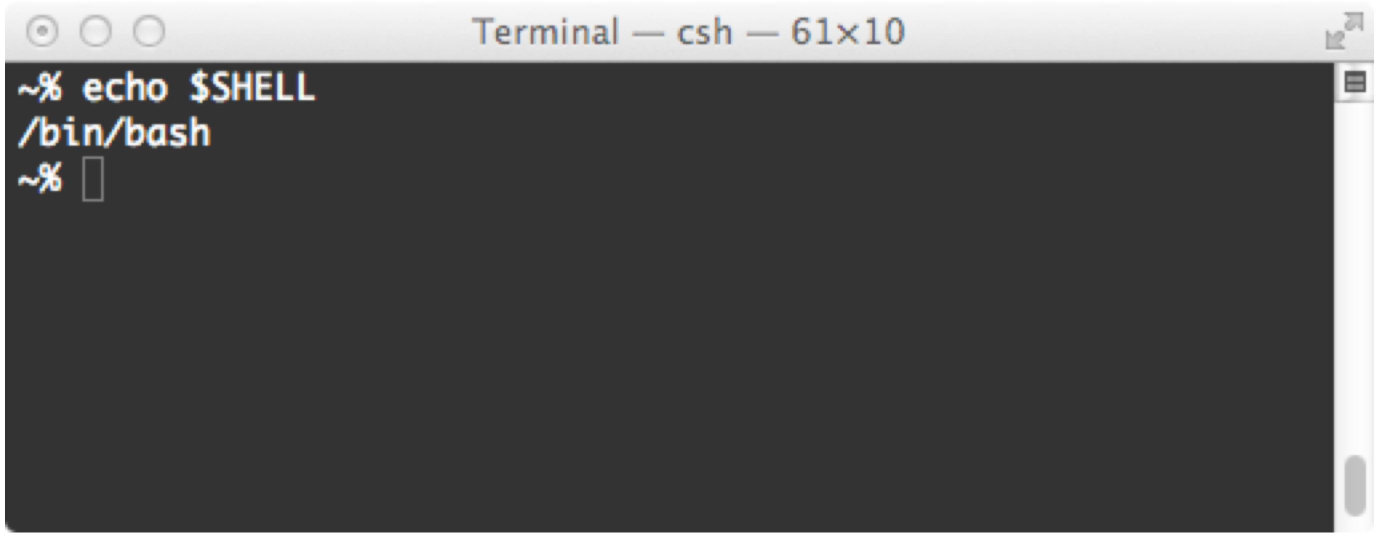
```
allegra1 /Applications/Microsoft Office 2011/Microsoft Word.app/Contents/MacOS $ ./'Microsoft Word'
```

The Terminal can be customised

- Change default bg color
- Change text size, colour and font
- Increase/decrease transparency
- Resize it
- Have multiple windows open side by side
- Have multiple "tabs" open at the same time
- Change the command prompt (most commonly a $ or % sign)
- Make the cursor blinking

The Unix shell

- The shell is a command-line interpreter that lets you interact with Unix
- The shell takes what you type and "decides" what to do with it
- The shell is actually a scripting language somewhat like Python
- It is always possible to change shell (either temporarily or permanently)



```
~% echo $SHELL
/bin/bash
~% 
```

The command-line interface (terminal) allows you:

- to send typed instructions to the computer (i.e., run programs, move/view files, etc.)
- to see the output that results from those instructions.

Every time you type any Unix command and **press enter**, the computer will attempt to follow your instructions and then, when finished, return you to the **command prompt**.

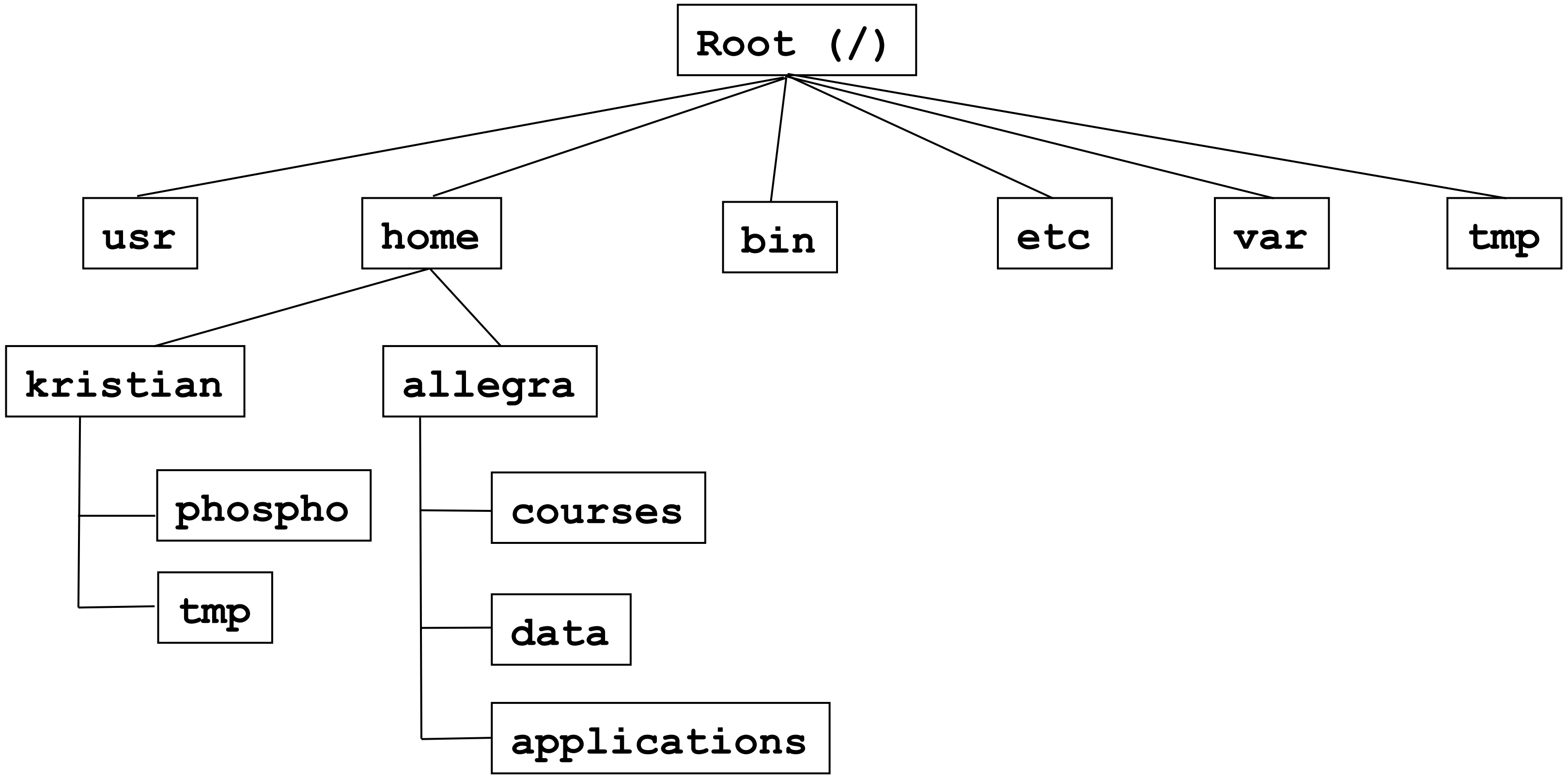
Type the Unix command ‘`ls`’ at the command prompt

What happens?

What is the filesystem tree?

The directory structure

The file-system is arranged in a hierarchical structure, like an inverted tree

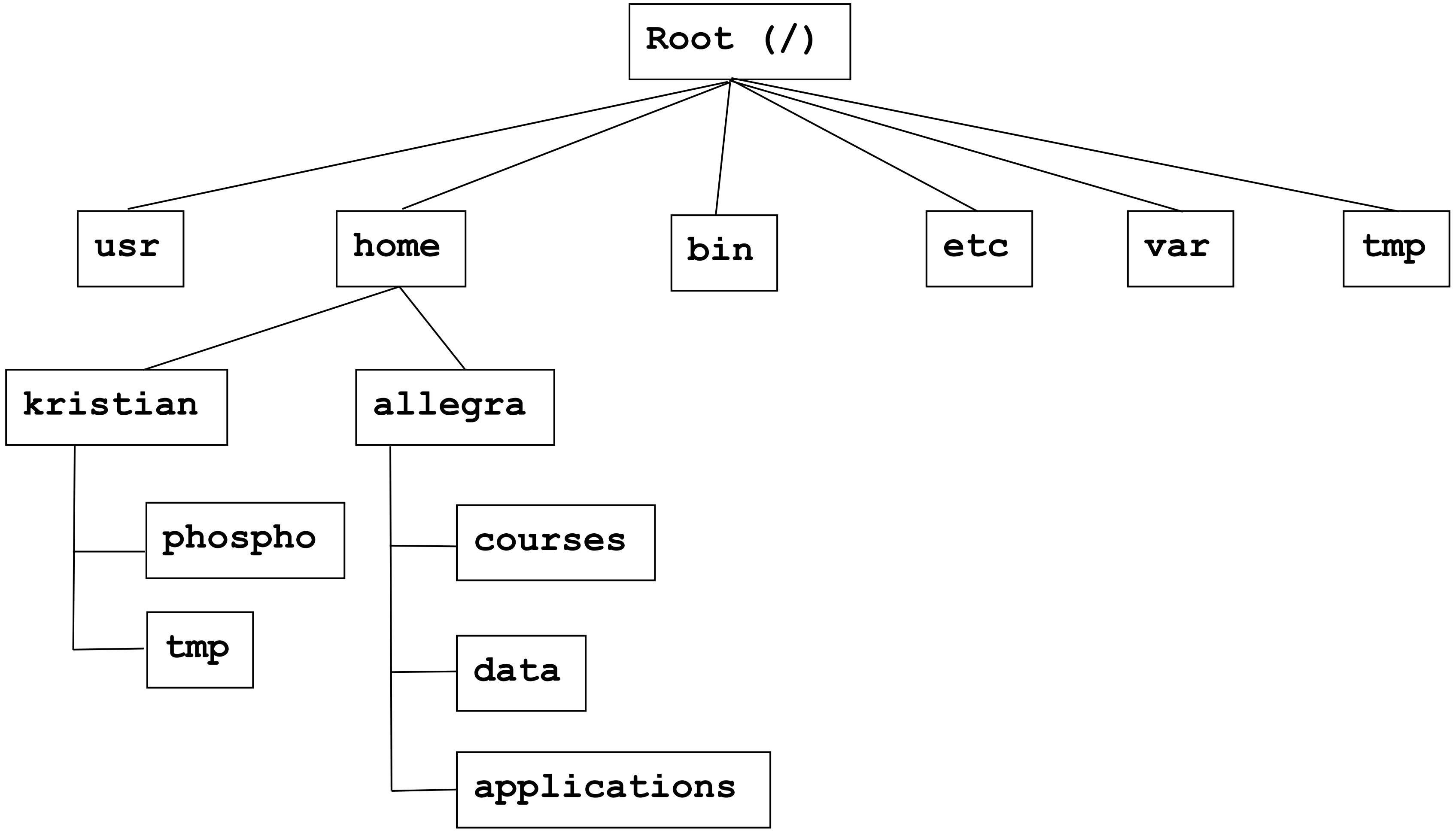


The top of the hierarchy is traditionally called **root**

When you first login, the current working directory is your home directory
(containing files and directories that only you can modify)

How can you navigate the filesystem?

What do you need to be able to do in order to navigate the filesystem?



In groups

What do you need to be able to do to navigate the filesystem?

- Find out where you are in the filesystem
- Change directory
- Find your way home
- Identify the location of a file/directory

What is the ***path*** of a file or a directory?

Slashes separate parts of the directory path:

```
/home/allegra/courses/TGAC2015/Academis_Linux.pdf
```

What do you need to be able to do in order to do/manage stuff in the filesystem?

- Think of things you need to be able to do in, e.g., Windows or Mac OSX

In groups

What do you need to be able to do in order to do/manage stuff in the filesystem?

- Think of things you need to be able to do in, e.g., Windows or Mac OSX
    - Make a new directory
    - Remove a directory
    - Copy a file to another file
    - Rename a file/directory
    - Create a file
    - Open/close a file
    - Remove a file
    - Run programs

What is a computer program?

Which ones do you know?

Did you know that…

…everything in Unix is either a file or a process?

A process is an **executing program** identified by a unique PID
(PID = Process IDentifier)

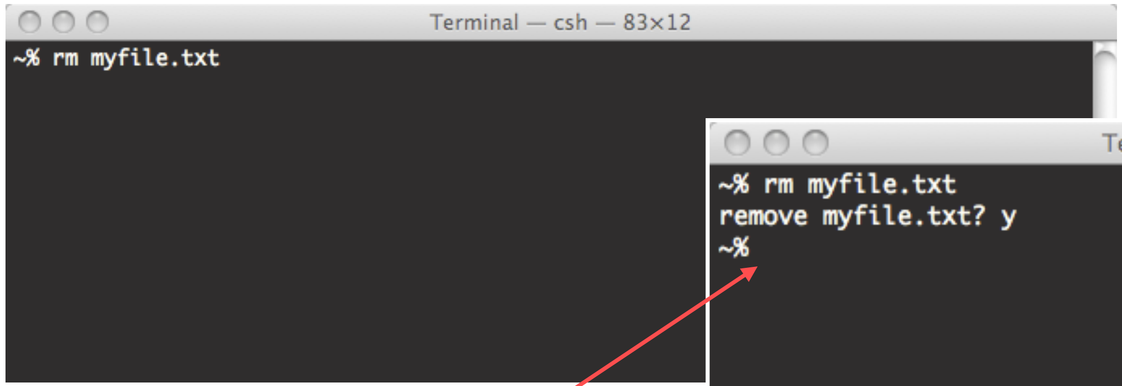
A file is a collection of data

About Unix commands

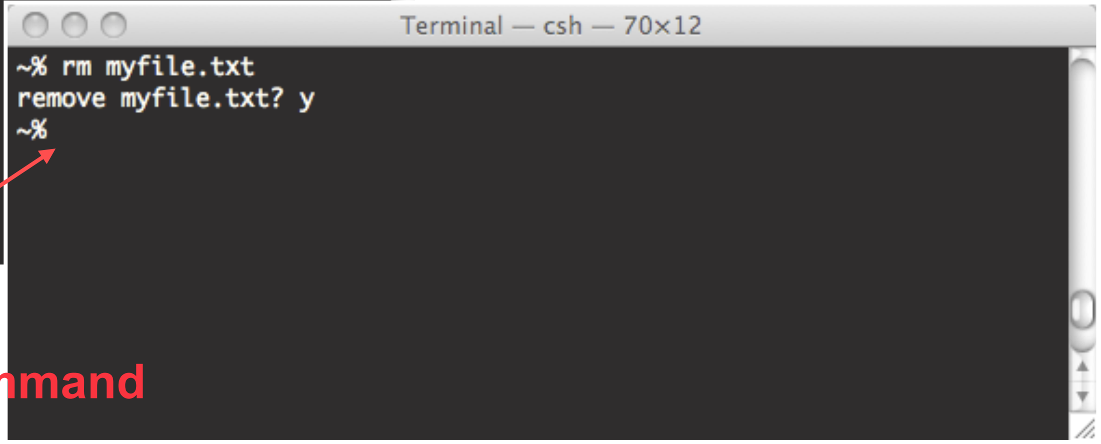
Commands are themselves programs

```
%rm myfile.txt [Return]
```

- The shell searches the file containing the program **rm**
- executes the program **`rm`** on **`myfile.txt`**
- After the process **`rm myfile.txt`** has finished running, the shell returns the prompt `%` to you, indicating that it is waiting for further commands.



```
~% rm myfile.txt
```



```
~% rm myfile.txt
remove myfile.txt? y
~%
```

Here you can type a new command

- `command_name -options <file> [Return]`

- `%ls [Return]`
- `%ls –l [Return]`
- `%ls -l <dirname> [Return]`
- `%ls -ltr <dirname> [Return]`

- `man <command name> [Enter]`
- `whatis <command name> [Enter]`

OPTIONS and ARGUMENTS

- There are commands that can take **XXX**
- Commands may also take **XXX**
- **XXX** change the behaviour of the command
- **XXX** are the objects on which commands act
- You will specify **XXX** using a **XXX**
- The command name, **XXX** and **XXX** must be separated by **XXX**

Replace the XXX

- If you've made a typo: Ctrl-**XXX** to cancel the whole line
- Unix is **XXX**-sensitive
- Ctrl-**XXX** sets the cursor at the beginning of the line
- Ctrl-**XXX** sets the cursor at the end of the line
- You can use up and down **XXX** to recall commands
- The command **XXX** tells you where is a given program
- You can use a **XXX** to write programs

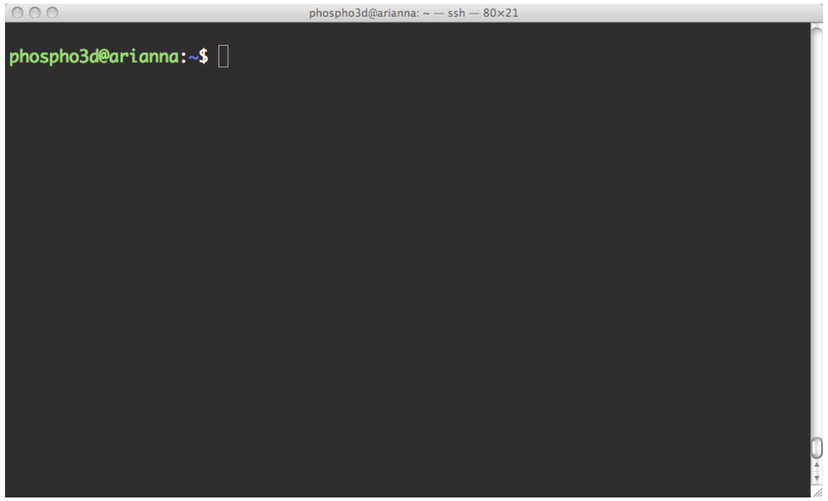
Writing and **running** programs in Unix

Where can we write programs?

What is a **text** editor?

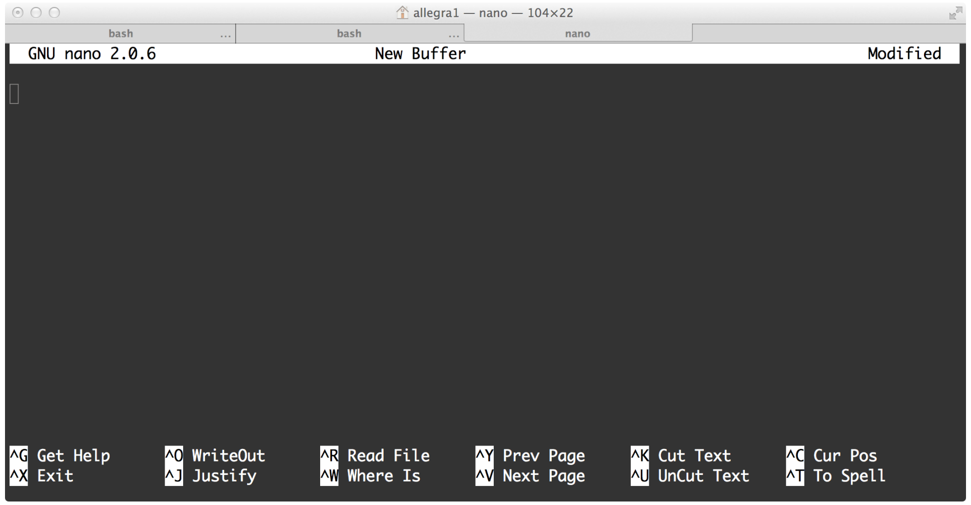
Which ones do you know?

- Access your home directory using the command-line interface



```
phospho3d@arianna:~$
```

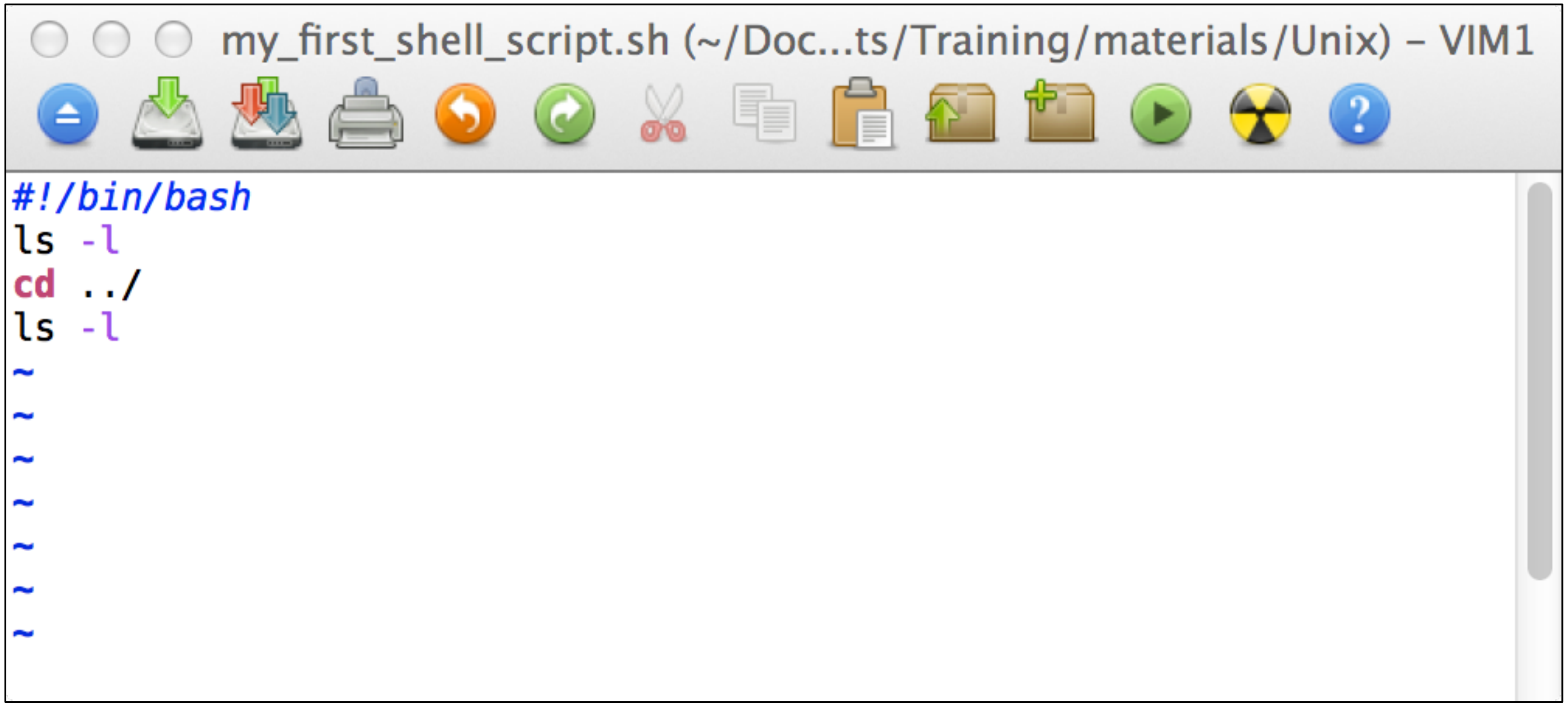
- Start the **nano** text editor



```
  GNU nano 2.0.6                    New Buffer                        Modified



^G Get Help   ^O WriteOut   ^R Read File  ^Y Prev Page  ^K Cut Text   ^C Cur Pos
^X Exit       ^J Justify    ^W Where Is   ^V Next Page  ^U UnCut Text ^T To Spell
```

- Create a text file “my_first_shell_script.sh”

My first shell script


my_first_shell_script.sh (~/Doc...ts/Training/materials/Unix) – VIM1

```
#!/bin/bash
ls -l
cd ../
ls -l
~
~
~
~
~
~
~
```

Write commands in a file, save and exit

- Go to the command-line interface and type “ls” at the prompt

How can we run programs on Unix?

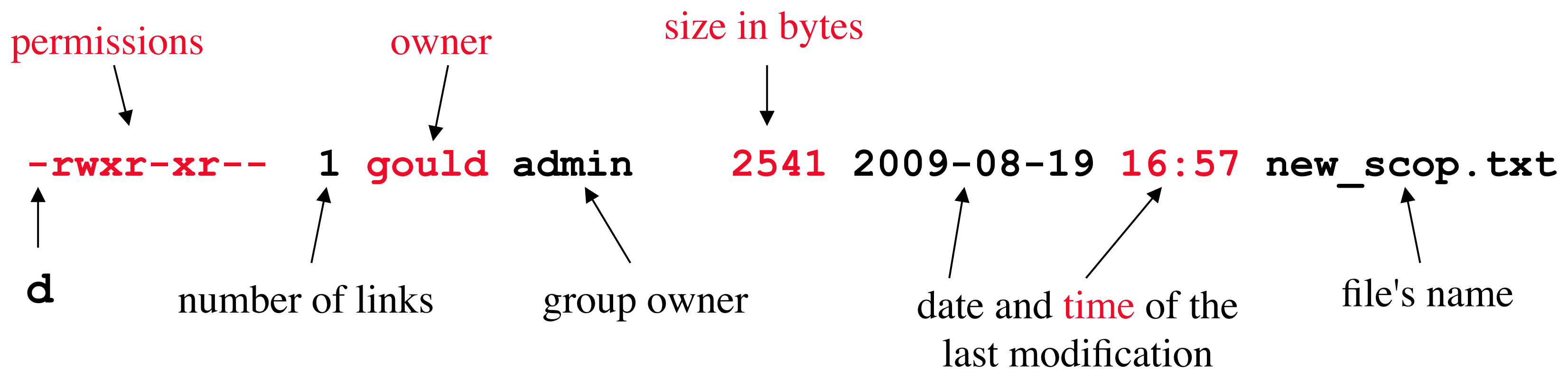
Prerequisites to run a program

1. The program must be somewhere on your computer
2. The program must be **executable**
3. You have to tell the **shell** which "**interpreter**" will read and execute the program AND where it will find it
4. You must be in the same directory as the program you want to run OR….
5. ….you can prefix its name with a path OR…
6. …the path to your program must in the **PATH environment variable**

Is my script executable?

File system security (access rights)

Each file (and directory) has associated access rights, which may be found by typing `ls -l`



```
-rwxr-xr--  1 gould admin    2541 2009-08-19 16:57 new_scop.txt
```

Access rights on directories

r allows users to list files in the directory
w allows users to delete files from the directory or move files into it
x allow users to access files in the directory

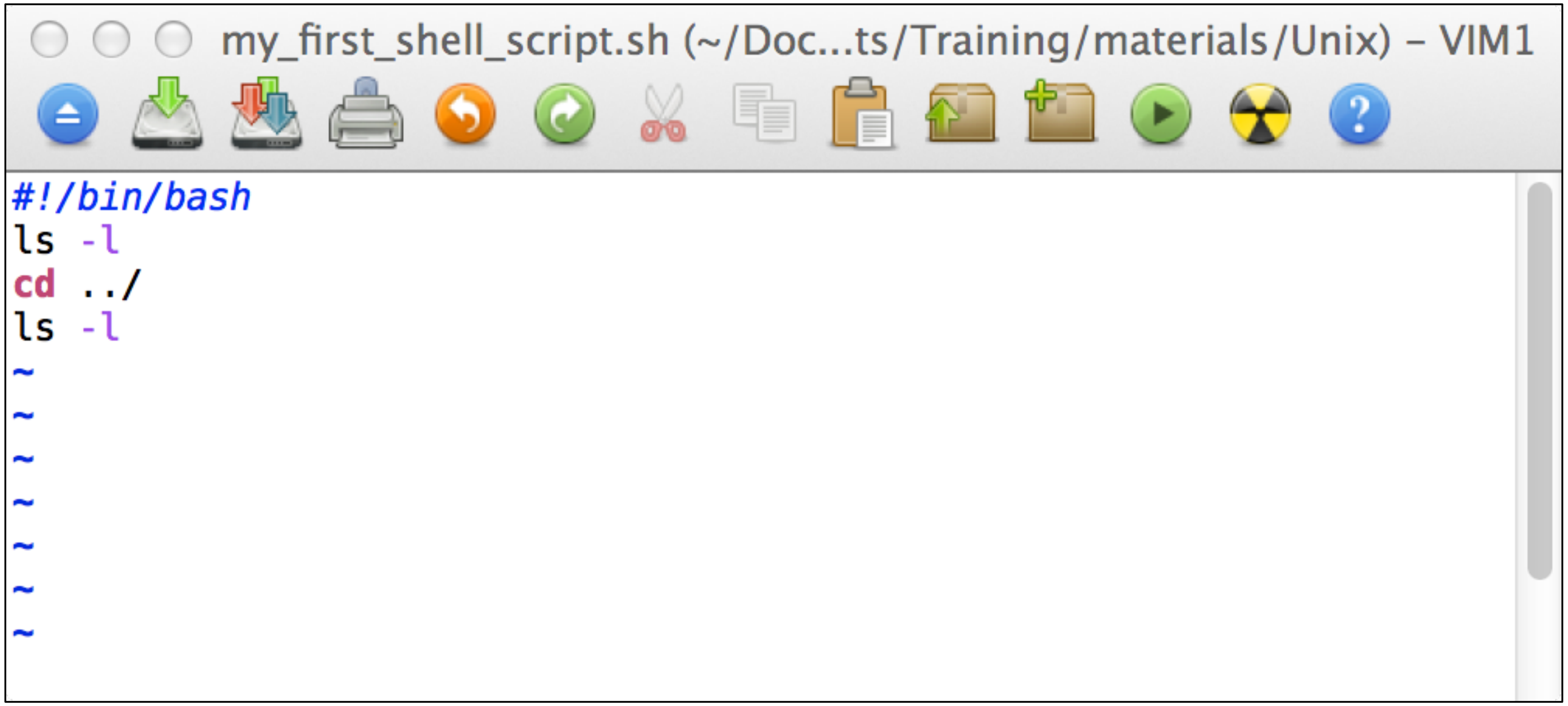
How can I make my script executable?

Changing access rights: chmod

```
%chmod go-rwx myfile.txt
%chmod a+x my_script
```

Symbol	Meaning
u	user
g	group
o	other
a	all
r	read
w	write (and delete)
x	execute (and access directory)
+	add permission
-	take away permission

You have to tell the **shell** which "**interpreter**" will read and execute the program AND where it will find it


my_first_shell_script.sh (~/Doc...ts/Training/materials/Unix) – VIM1

```
#!/bin/bash
ls -l
cd ../
ls -l
~
~
~
~
~
~
~
```

`#!/bin/bash`

"Aha, you want to use the program located at `/bin/bash` to interpret all the instructions that follow"

Now you want to execute the script

You have to tell Unix where it can find it

Where Unix searches for programs?

Where Unix searches for programs?

Once you have made a script executable you can always run it by prefixing its name with a path:

```
./shell_commands.sh

~allegra/Documents/shell_commands.sh
```

- Anytime you are running a program, Unix will check through **a list of predefined directories** to see if that program exists in any of those locations.
- If it finds a match, it will try running the program and stop looking in any other directory.
- If it cannot find a match, it will print "`command not found`"

UNIX environment variables

Unix keeps track of several special variables that are associated with your account

- Written in upper-case letters
- Start with a $
- `echo $SHELL`
- `printenv SHELL`
- `echo $PATH`

`echo $PATH`

If the system returns a message saying "**`command: Command not found`**", this indicates that either the command doesn't exist at all on the system or it is simply not in your path.

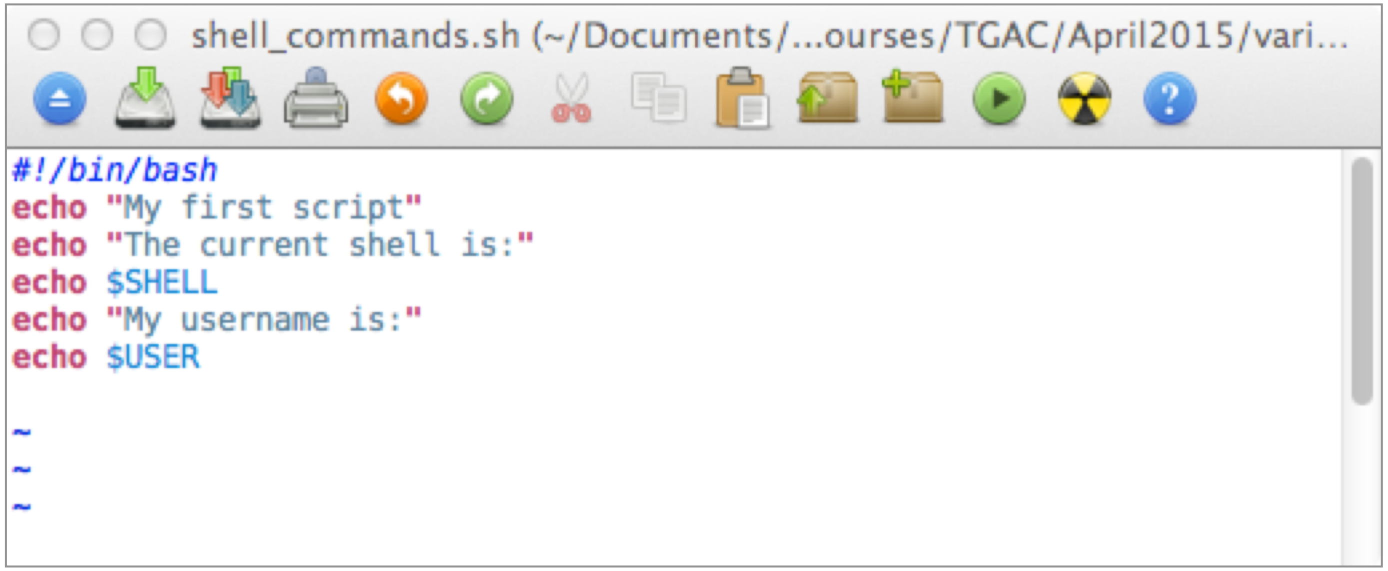
```
# for shells in the bash family
export PATH=$PATH:~/allegra/my_scripts

# for shells in the csh family
setenv PATH $PATH\:~/allegra/my_scripts
```

- Any program in `~/allegra/my_scripts` can be run from **anywhere** in the filesystem (as long as the program file is executable)
- You can use tab-completion
- Your scripts will be treated like any Unix command

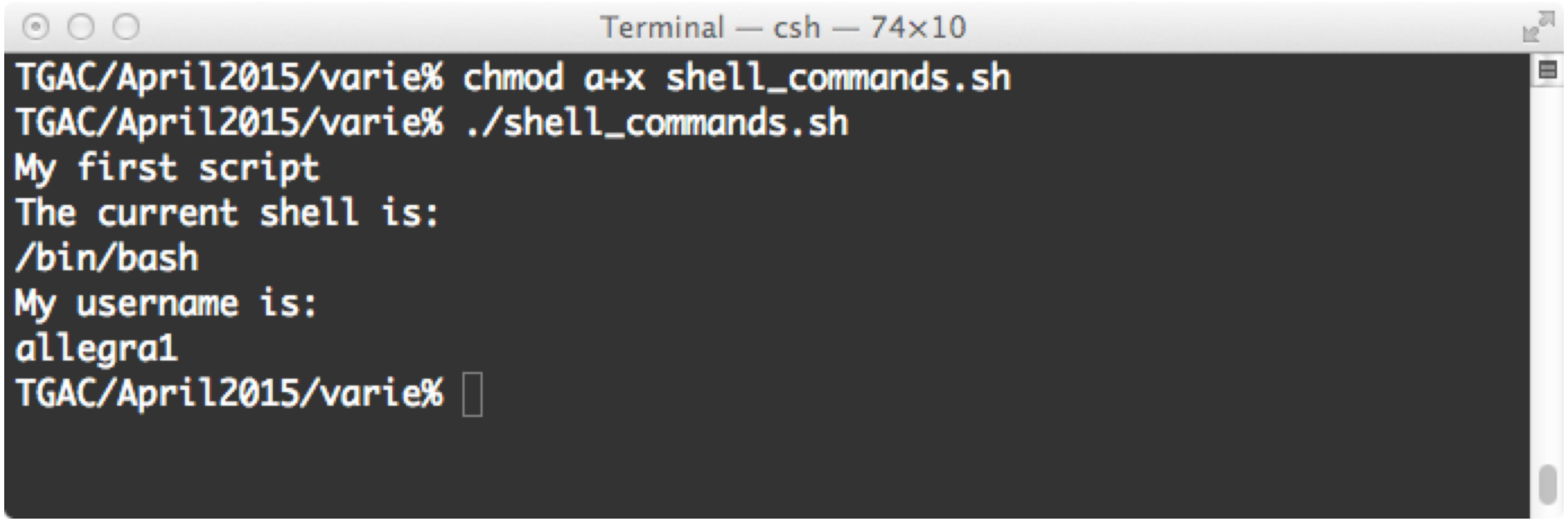
A few more questions…

- What is command-line completion?
- What is a default argument?

Exercise: use a text editor to write commands into a file, save, exit, make it executable and run it


shell_commands.sh (~/Documents/...ourses/TGAC/April2015/vari...

```
#!/bin/bash
echo "My first script"
echo "The current shell is:"
echo $SHELL
echo "My username is:"
echo $USER

~
~
~
```



```
TGAC/April2015/varie% chmod a+x shell_commands.sh
TGAC/April2015/varie% ./shell_commands.sh
My first script
The current shell is:
/bin/bash
My username is:
allegra1
TGAC/April2015/varie%
```

Connecting to a remote computer

```
ssh remote_host
```

The *remote_host* is the IP address or domain name that you are trying to connect to.

If your username is different on the remote system:

```
ssh remote_username@remote_host
```

Once you have connected to the server, you will probably be asked to verify your identity by providing a password.

```
ssh -x remote_username@remote_host
```

Transferring files to/from a remote computer

```
sftp username@host
```

Enter your password when prompted
Several Unix commands do work
`get` → Copy a file from the remote computer to the local computer.
`put` → Copy a file from the local computer to the remote computer.

Transferring files to/from a remote computer

scp copies files over a secure, encrypted network connection.

```
scp /home/image*.jpg allegra@myhost.com:/home/images
scp allegra@myhost.com:/home/image*.jpg /home/allegra/downloads
```

```
scp [-12346BCpqrv] [-c cipher] [-F ssh_config] [-i identity_file]
    [-l limit] [-o ssh_option] [-P port] [-S program]
    [[user@]host1:]file1 ... [[user@]host2:]file2
```

Enter your password when prompted

Non-interactive download of files from the Web

```
wget [option]... [URL]...
```

- Non-interactive means that it can work in the background, while the user is not logged on.
- This allows you to start a retrieval and disconnect from the system, letting Wget finish the work.
- By contrast, most of the Web browsers require constant user's presence, which can be a great hindrance when transferring a lot of data.

Listing files and directories

`ls`	list files and directories
`ls -a`	list all files and directories
`mkdir`	make a directory
`cd` *`directory`*	change to named directory
`cd`	change to home-directory
`cd ~`	change to home-directory
`cd ..`	change to parent directory
`pwd`	display the path of the current directory

The directories '.', '..', and '~'

```
% ls -a [Enter]

% cd . [Enter]

% cd .. [Enter]

% ls ~/oeiras
```

Handling files and directories

`cp` *file1 file2*	copy file1 and call it file2
`mv` *file1 file2*	move or rename file1 to file2
`rm` *file*	remove a file
`rmdir` *directory*	remove a directory
`cat` *file*	display a file
`more` *file*	display a file a page at a time
`head` *file*	display the first few lines of a file
`tail` *file*	display the last few lines of a file
`grep` *'keyword' file*	search a file for keywords
`wc` *file*	count number of lines/words/characters in file

more **less** **clear**

Redirection

command > file	redirect standard output to a file
command >> file	append standard output to a file
command < file	redirect standard input from a file
cat *file1 file2 > file0*	concatenate file1 and file2 to file0
sort	sort data
who	list users currently logged in