# Table of Contents



# Introduction to Unix, Part 1 - Intro to Unix and Terminal

**Welcome to the Programming for Evolutionary Biology workshop!!**

Giovanni M. Dall'Olio and Alvaro Perdomo-Sabogal, 03/03/2019. All materials available here:
https://github.com/dalloliogm/evop2019/archive/master.zip
(https://github.com/dalloliogm/evop2019/archive/master.zip)

In this first part of the course we will see the basics of Unix and Terminal usage.

**How to use these slides**: Press Space to get to the next slide. Use arrows to navigate the subsections.

# What is the command line and why should we use it?

Most computations in bioinformatics and data science are done from a command line interface, rather than from a point-and-click approach.

This might seem less intuitive and quite cumbersome at the beginning, but it is actually very powerful and direct.

Today we will make our first steps with the command line, learn the history and approach of the Unix systems, and see what are the advantages of this approach.

# What is Unix?

First of all, let's clarify some definitions.

**Unix** is the name of an operating system created in the '80s, which became popular for a variety of reasons:

- it introduced a novel approach to programming and file managing
- it runs on any computer hardware independently of the maker
- it was free for academic use

## Linux, MacOs, and Unix

The original Unix system does not run on modern computers, but we can use some of its descendants (Linux, Solaris, MacOs)


# The Unix Philosophy

The Unix philosophy can be summarised as:

- Make each program do one thing well.
- Expect the output of every program to become the input to another, as yet unknown, program.
- Work on file streams, reading one line at a time.

Press Space to continue.

We will see how each Unix tool is specialized on a single task, and how the piping system allows to combine these tool together.

These principles can be useful to any person wishing to learn programming. You may use the same approach when learning programming, starting writing small programs and functions, and combining them together in bigger pipelines.

# TAs and support

Some practical information:

- There are six Teaching Assistants (TAs), to help you, approximately one every five students.
- Use sticky notes during the exercises: a green note when you finished the exercise, and a red note if you need help
- We will have a second projector where a TA will shows the commands to type

# Hands on Linux: let's login!

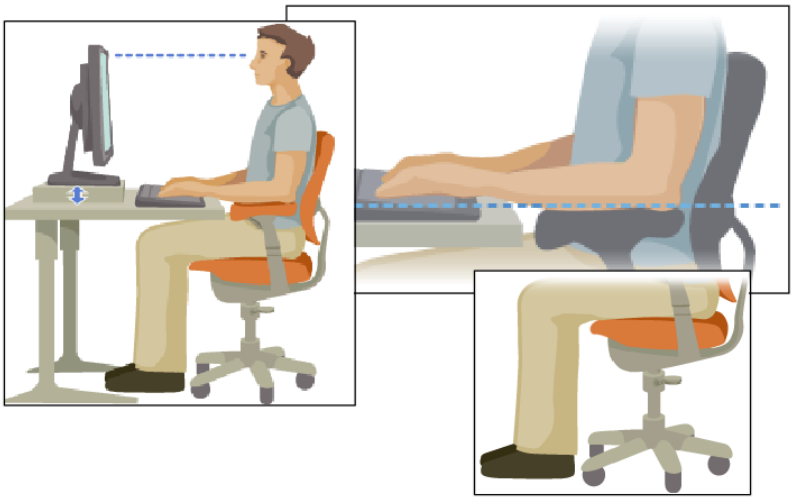
During this course we will set two sets of computers:

- the **classroom** computers (in front of you)
- **one** login server this year: **'evop2018login.imp.fu-berlin.de'** .

Let's focus on the classroom computers for now.

Check the **login information: Password and Computer infrastructure** that Katja provided you with.

## While the computer loads: make yourself comfortable!



## This is how the desktop should look like, after login

Once logged in, you will see a desktop very similar to a Windows or Mac environment:

(The wallpaper and desktop may be different)

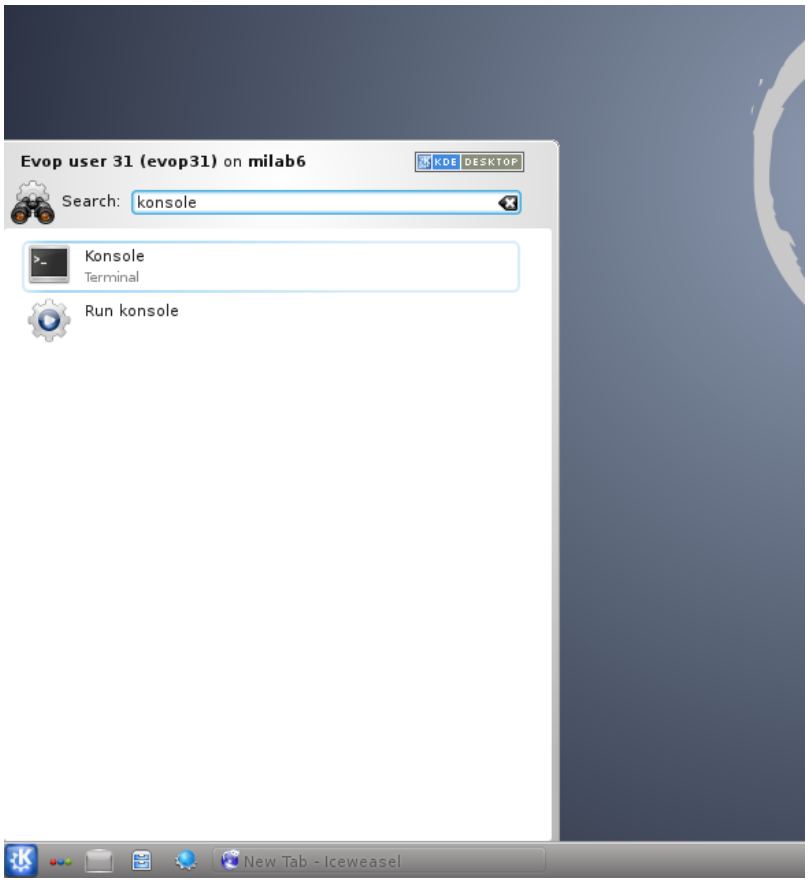
## The Linux desktop

The Linux interface may be different to Windows and Mac, but it should not be difficult to use:

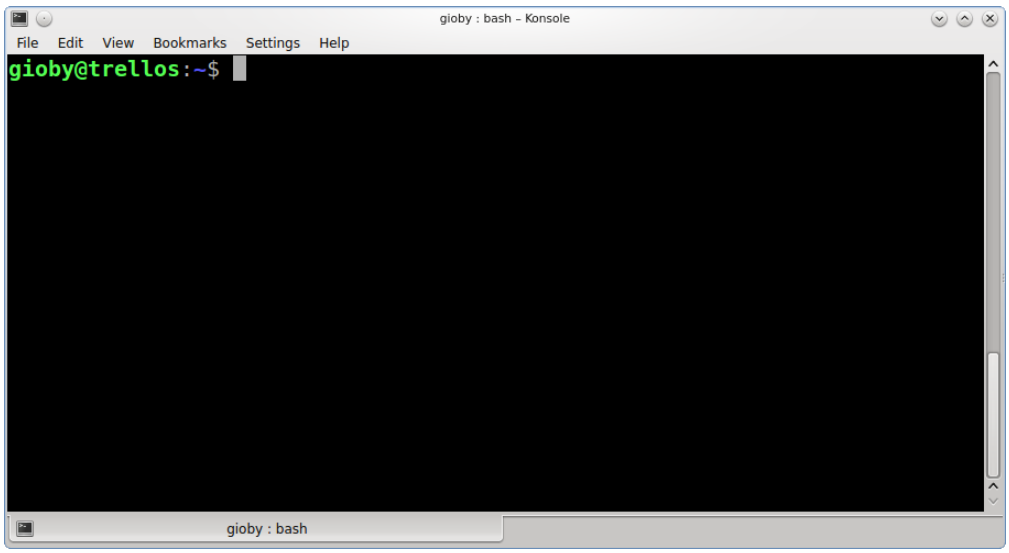
- you have a Launch menu, from where you can access all the applications installed
- most applications are similar:
  - firefox or chrome for web browsing
  - kate, gedit for editing text
  - libreoffice for documents, presentation, spreadsheet

# Let's open the terminal

Open the Launch menu and search for "terminal" or "konsole"




# How does a terminal looks like?




# More definitions

**Linux**:

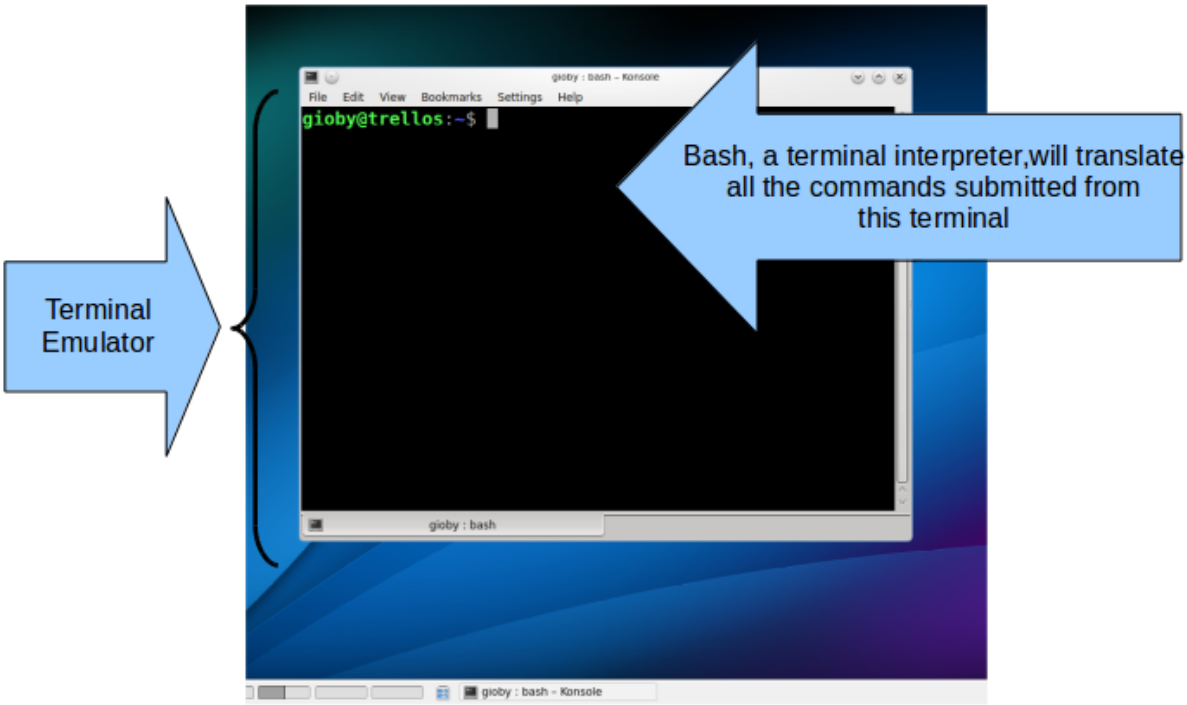
```
A "descendant" of Unix, e.g. an operating system based on Unix that can run
on modern computers
```

**Terminal**:

```
A software that allows to input commands to the computer, by typing them ra
ther than point-and-click
```

**Bash**:

```
A command-line interpreter, e.g. a software that interprets the commands gi
ven from the terminal, and execute them.
```




# Launching applications from the Terminal

The terminal can be used any application installed in the computer, by typing its name instead of clicking on it.

For example, type *firefox* to launch a web browser:

```
$: firefox
```

* if this doesn't work, try: google-chrome

# Running applications in the background

Some applications "freeze" the terminal when executed. For example, let's launch *kate*, a text editor

```
$: kate
```

* if this doesn't work, try *gedit*

There are two alternative solutions:

- after launching the application, press CTRL-z, then run the bg command
- for future cases, before launching the application, add an & after the command ( kate & )

You can also type *jobs* for a list of all the application running in background.

## Killing an application

Sometimes an application gets stuck and doesn't respond to input.

Or for some reasons we want to abort the execution of a long script, because we realize there is an error in the code.

If the application is running in the foreground, we can press CTRL-c to abort its execution.

If the application is in the background (previous slide), we can use:

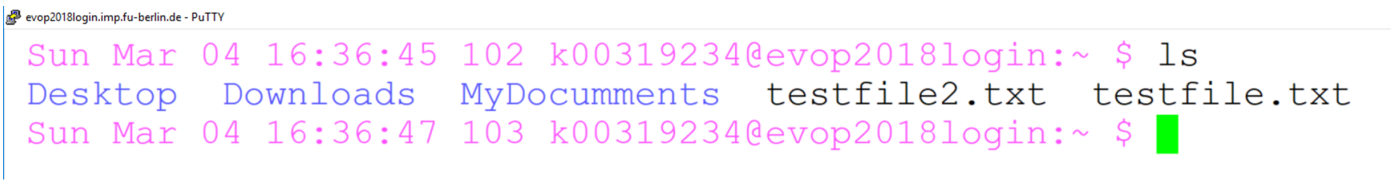
- jobs (to list all the jobs in background)
- kill %1 (to kill the first job in background)

# Your first command: ls

The most basic command is **ls**, which allows to list all the files in the current folder.

The name is an abbreviation for "list"

Typing **ls** will show all the files in the current directory.




Press space or the down key to continue.

## Anatomy of a command

A command can be composed by three parts:

- the command itself
- parameters (to alter the default behaviour of the command)
- arguments (target of the command)

## listing all files and their details (ls -l)

You can use the -l parameter of ls to visualize more details:

```
-bash-4.3$ ls -l
total 28
drwxr-xr-x 2 alvaro staff 4096 Feb 26 10:57 Desktop
drwxr-xr-x 2 alvaro staff 4096 Feb 26 10:57 Documents
drwxr-xr-x 2 alvaro staff 4096 Feb 26 10:57 Downloads
-rw-r--r-- 1 alvaro staff    1 Feb 26 10:58 file.txt
drwxr-xr-x 2 alvaro staff 4096 Feb 26 10:57 Images
drwxr-xr-x 2 alvaro staff 4096 Feb 26 10:57 Templates
-rw-r--r-- 1 alvaro staff    1 Feb 26 10:58 test1.txt
-bash-4.3$
```

## Listing hidden files (ls -a; ls -la)

By convention, in Unix systems hidden files have a name beginning with a "."

The -a option of ls shows all the files, including the hidden ones.

In general your home folder contains several hidden configuration files (.e.g. .bashrc, .conf, .cache). We can ignore them for the moment.

```
-bash-4.3$ ls -la
total 36
drwxr-xr-x  7 alvaro staff 4096 Feb 26 10:58 .
drwxr-xr-x 85 alvaro staff 4096 Feb 26 15:18 ..
drwxr-xr-x  2 alvaro staff 4096 Feb 26 10:57 Desktop
drwxr-xr-x  2 alvaro staff 4096 Feb 26 10:57 Documents
drwxr-xr-x  2 alvaro staff 4096 Feb 26 10:57 Downloads
-rw-r--r--  1 alvaro staff    1 Feb 26 10:58 file.txt
drwxr-xr-x  2 alvaro staff 4096 Feb 26 10:57 Images
drwxr-xr-x  2 alvaro staff 4096 Feb 26 10:57 Templates
-rw-r--r--  1 alvaro staff    1 Feb 26 10:58 test1.txt
-bash-4.3$
```

## Combining multiple ls parameters together

We can combine multiple parameters of a command line together.

For example, by typing "ls -lt", we can show the files in the long format (-l), and sorted by modification date (-t)

```
-bash-4.3$ ls -lt
total 28
-rw-r--r-- 1 alvaro staff    1 Feb 26 10:58 test1.txt
-rw-r--r-- 1 alvaro staff    1 Feb 26 10:58 file.txt
drwxr-xr-x 2 alvaro staff 4096 Feb 26 10:57 Templates
drwxr-xr-x 2 alvaro staff 4096 Feb 26 10:57 Images
drwxr-xr-x 2 alvaro staff 4096 Feb 26 10:57 Downloads
drwxr-xr-x 2 alvaro staff 4096 Feb 26 10:57 Desktop
drwxr-xr-x 2 alvaro staff 4096 Feb 26 10:57 Documents
-bash-4.3$
```

# File permissions and output of ls -l

The first column in the ls -l output describes the permissions for the file (e.g. drwxr-xr-x.)

- character **1**: whether the element is a directory (d) or a file (-)
- characters **2-4**: whether the file is readable (r), writeable (w), or executable (x) by the user who owns it (fedora)
- characters **5-7**: the same, but for the user group (also called fedora)
- characters **8-10**: the same, for any user
- character **11**: security settings (not in all systems)

See **chmod** to modify file permissions.

```
-bash-4.3$ ls -l
total 28
drwxr-xr-x 2 alvaro staff 4096 Feb 26 10:57 Desktop
drwxr-xr-x 2 alvaro staff 4096 Feb 26 10:57 Documents
drwxr-xr-x 2 alvaro staff 4096 Feb 26 10:57 Downloads
-rw-r--r-- 1 alvaro staff    1 Feb 26 10:58 file.txt
drwxr-xr-x 2 alvaro staff 4096 Feb 26 10:57 Images
drwxr-xr-x 2 alvaro staff 4096 Feb 26 10:57 Templates
-rw-r--r-- 1 alvaro staff    1 Feb 26 10:58 test1.txt
-bash-4.3$
```

## Arguments

The other component of a command is the argument, e.g. what is the target of the command.

When we type **ls**, by default the target of the command is the current directory (also referred as ".")

To see the contents of another directory, just add its name to the command invocation:

```
ls -l Desktop
```

## Command-line tip: the Tab key and auto-completion

We can use the Tab key on the keyboard (usually above Caps Lock) to automatically complete a command or a directory name.

For example, try the following:

```
ls Desk<press Tab key>
```

# How to get documentation of a command?

How can we know all the possible options of a command?

There are at least three options:

- `--help`
- `man`
- `info`

## ls --help

Most command support a "--help" option (or -h, -help), which prints a summary of the most common parameters and options for the command:

```
$: ls --help

Usage: ls [OPTION]... [FILE]...
List information about the FILEs (the current directory by default).
Sort entries alphabetically if none of -cftuvSUX nor --sort is specifie
d.

Mandatory arguments to long options are mandatory for short options to
o.
  -a, --all                  do not ignore entries starting with .
  -A, --almost-all           do not list implied . and ..
      --author               with -l, print the author of each file
  -b, --escape               print C-style escapes for nongraphic chara
cters
      --block-size=SIZE      scale sizes by SIZE before printing them;
 e.g.,
```

## The man page of a command

The **man** command allows to access the manual of a command.

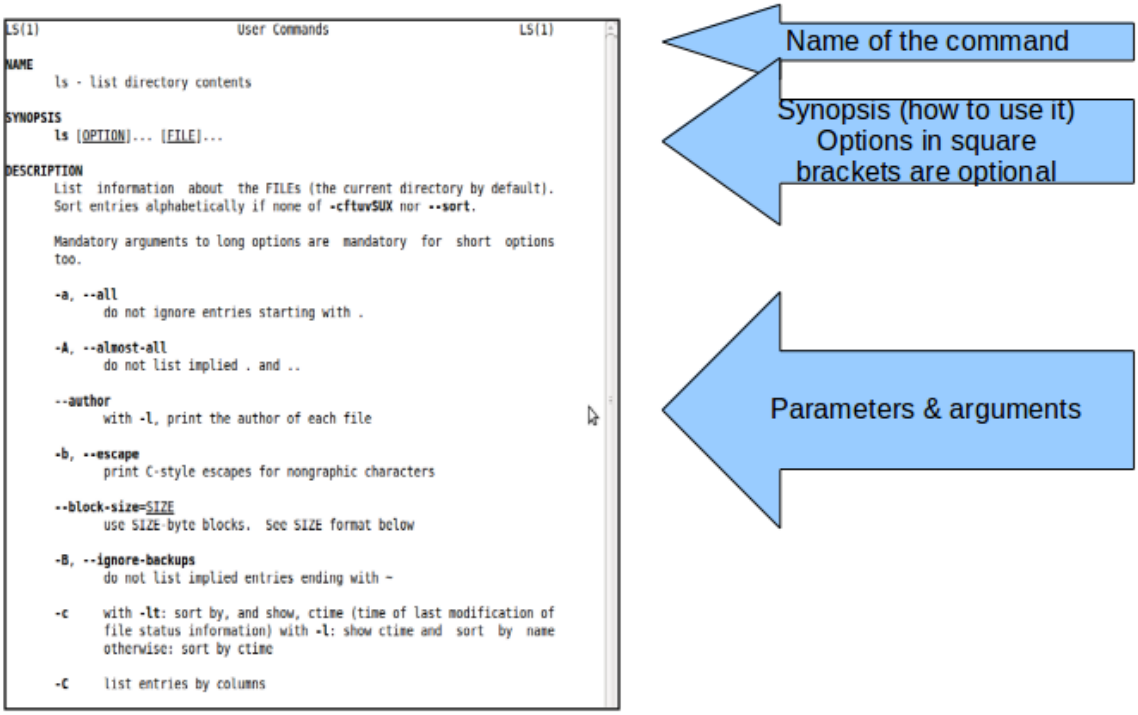
Let's try it:

```
man ls
```

## The man page of ls

Most man pages contain at least the following sections:

- NAME: name of the command
- SYNOPSIS: how to use it
- DESCRIPTION/OPTIONS: what the command does, and


LS(1) User Commands LS(1)
NAME
ls - list directory contents
SYNOPSIS
ls [OPTION]... [FILE]...
DESCRIPTION
List information about the FILEs (the current directory by default).
Sort entries alphabetically if none of -cftuvSUX nor --sort.
Mandatory arguments to long options are mandatory for short options
too.
-a, --all
do not ignore entries starting with .
-A, --almost-all
do not list implied . and ..
--author
with -l, print the author of each file
-b, --escape
print C-style escapes for nongraphic characters
--block-size=SIZE
use SIZE-byte blocks. See SIZE format below
-B, --ignore-backups
do not list implied entries ending with ~
-c with -lt: sort by, and show, ctime (time of last modification of
file status information) with -l: show ctime and sort by name
otherwise: sort by ctime
-C list entries by columns
Name of the command
Synopsis (how to use it)
Options in square
brackets are optional
Parameters & arguments

In [16]:

```
man ls | head -n 30
```

```
LS(1)                            User Commands
LS(1)



NAME
       ls - list directory contents

SYNOPSIS
       ls [OPTION]... [FILE]...

DESCRIPTION
       List  information  about  the FILEs (the current directory by d
efault).
       Sort entries alphabetically if none of -cftuvSUX nor --sort  is
speci-
       fied.

       Mandatory  arguments  to  long  options are mandatory for short
options
       too.

       -a, --all
              do not ignore entries starting with .

       -A, --almost-all
              do not list implied . and ..

       --author
              with -l, print the author of each file

       -b, --escape
              print C-style escapes for nongraphic characters
```

## Navigating a man page

- Use the arrows keys or PageUp/PageDown to scroll the man page
- Use "/" followed by a keyword to search text
  - e.g. press /sort to search for the word "sort"
- press "q" to exit

## Other useful sections in a man page

- SEE ALSO: references to similar commands
- EXAMPLES: examples of how to use the command

```
File  Edit  View  Search  Terminal  Help

SEE ALSO
   Regular Manual Pages
       awk(1),   cmp(1),   diff(1),  find(1),  gzip(1),  perl(1),  sed(1),
       sort(1),   xargs(1),   zgrep(1),   mmap(2),    read(2),    pcre(3),
       pcresyntax(3), pcrepattern(3), terminfo(5), glob(7), regex(7).

   POSIX Programmer's Manual Page
       grep(1p).

   TeXinfo Documentation
       The  full documentation for grep is maintained as a TeXinfo manual.
       If the info and grep programs are properly installed at your  site,
       the command

              info grep

       should give you access to the complete manual.

NOTES
       GNU's not Unix, but Unix is a beast; its plural form is Unixen.

 Manual page grep(1) line 640/664 100% (press h for help or q to quit)
```

```
File  Modifica  Visualizza  Cerca  Terminale  Aiuto

       --version
              output version information and exit

       Note, comparisons honor the rules specified by `LC_COLLATE'.

EXAMPLES
       comm -12 file1 file2
              Print only lines present in both file1 and file2.

       comm -3
              file1 file2  Print lines in file1 not in file2, and vice versa.

AUTHOR
       Written by Richard M. Stallman and David MacKenzie.

REPORTING BUGS
       Report comm bugs to bug-coreutils@gnu.org
-- MOST: *stdin*                                                (33,1) 43%
Press `Q' to quit, `H' for help, and SPACE to scroll.
```

# Getting documentation via info

The command "info" allows to access a more complete description of a command.

```
File  Edit  View  Search  Terminal  Help
File: coreutils.info,  Node: ls invocation,  Next: dir invocation,  Up: Dir\
ectory listing

10.1 `ls': List directory contents
==================================

The `ls' program lists information about files (of any type, including
directories).  Options and file arguments can be intermixed
arbitrarily, as usual.

   For non-option command-line arguments that are directories, by
default `ls' lists the contents of directories, not recursively, and
omitting files with names beginning with `.'.  For other non-option
arguments, by default `ls' lists just the file name.  If no non-option
argument is specified, `ls' operates on the current directory, acting
as if it had been invoked with a single argument of `.'.

   By default, the output is sorted alphabetically, according to the
locale settings in effect.(1) If standard output is a terminal, the
output is in columns (sorted vertically) and control characters are
output as question marks; otherwise, the output is listed one per line
--zz-Info: (coreutils.info.gz)ls invocation, 58 lines --Top-----------------
```

Not all commands have an info page, though.

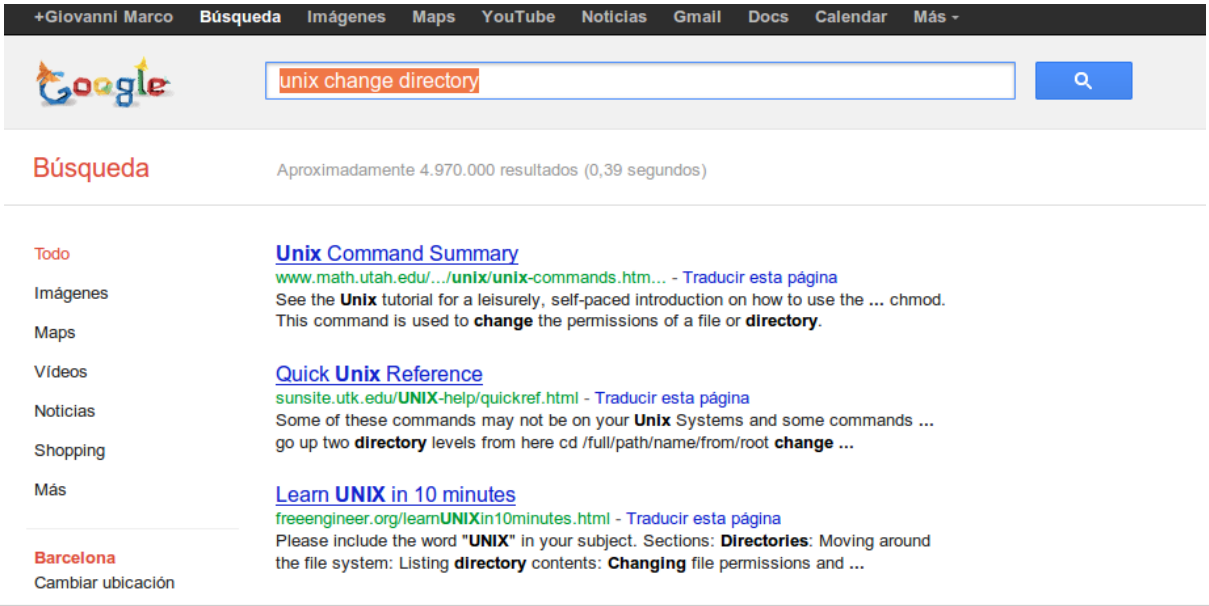
# Quick exercise

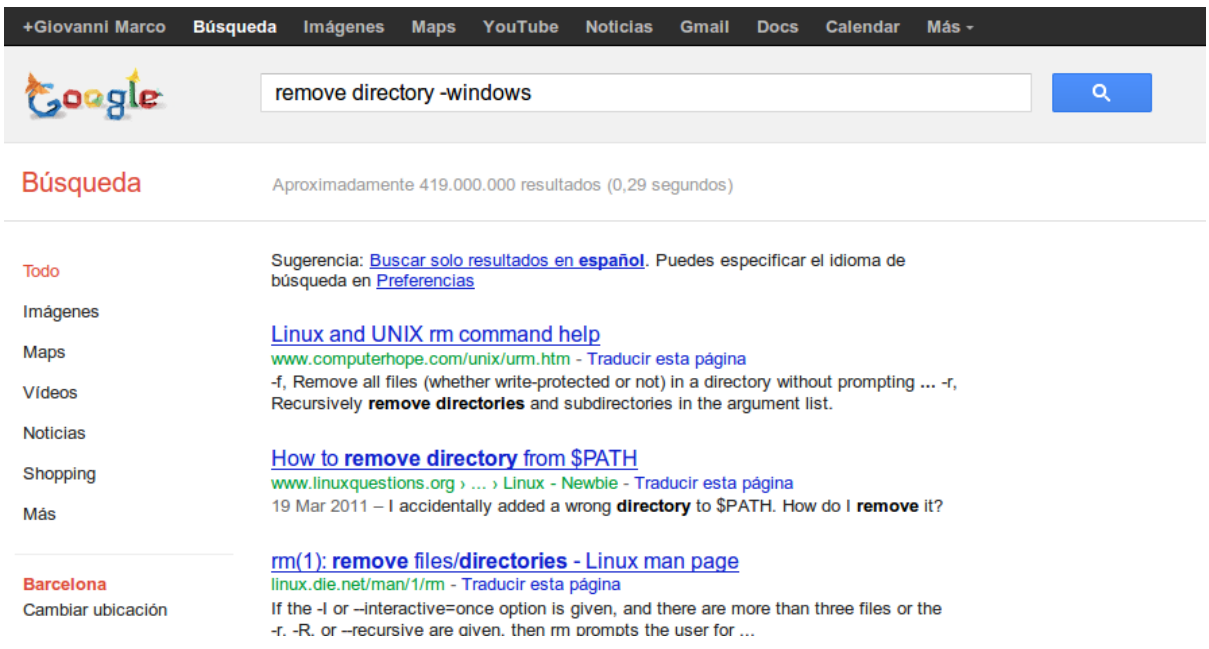
- Which parameter can be passed to **ls** to sort files by size?
- How to show the contents of a directory recursively with **ls?**

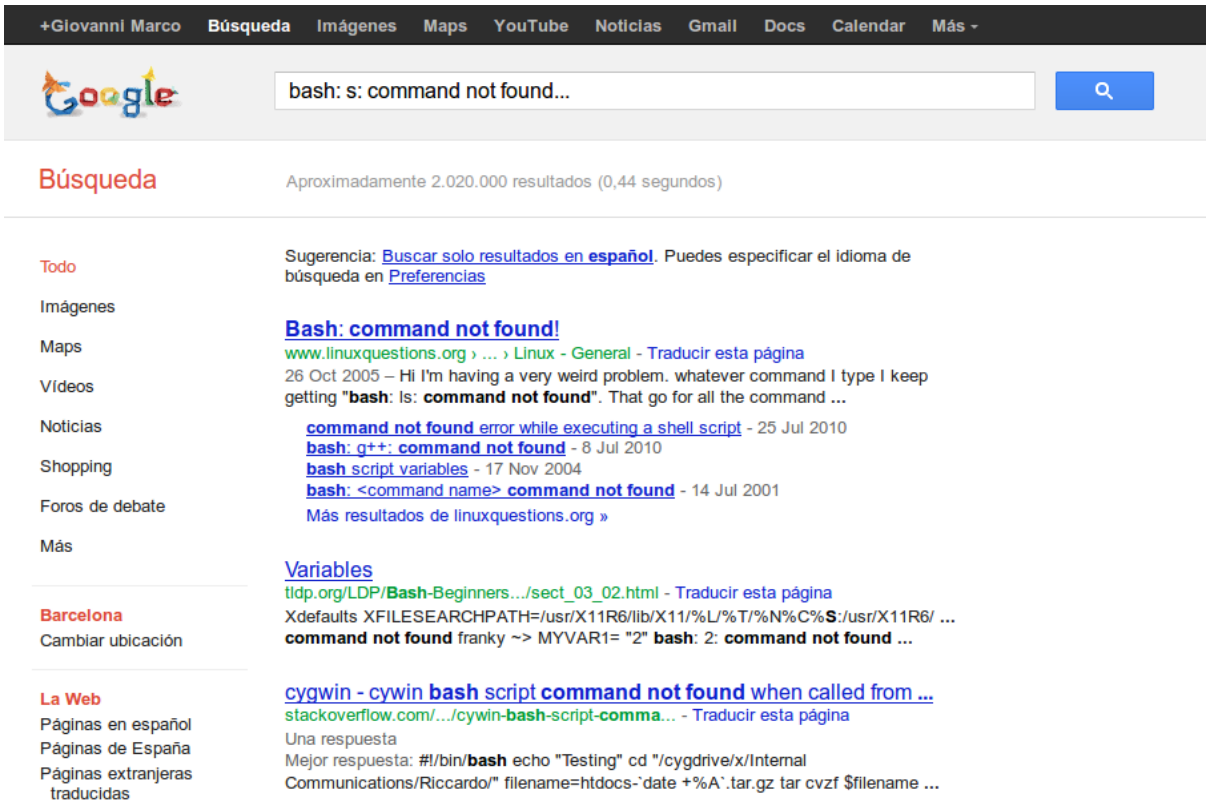
# Getting help from Internet

Some tips to get better results when searching the documentation of a command on Internet:

- add keywords such as "unix", "bash", "fedora"
- Use the "-" operator to remove junk results
- if you get an error with a softawre or during an installation, copy and paste the message on google






# How can I use a Unix system at home/lab?

## The Linux distributions

One option to access a Unix system is to install a "GNU/Linux" operating system on your computer.

However there is not a single "GNU/Linux" operating system, as different groups of people have created different distributions according to different needs or contexts

## What is the difference between Linux Distributions?

- The software included by default when you install the system
  - e.g. firefox or chrome, gedit or kate
- Some distributions include only free software, other are less strict
- Some distributions are aimed for new users, while other are designed for "nerds, geeks, specialists....."

Behind the scenes, distributions can differ for:

- the system used to install new software
- technical details such as libraries used to compile the software

# Examples of Linux Distributions

- Ubuntu: a popular distribution aimed at beginners
- BioLinux: a derivation of Ubuntu, including software for bioinformaticians
- Fedora: another general use distribution

### BioLinux

BioLinux is a distribution designed for bioinformaticians

- Blast
- bioperl
- vcftools, samtools, ...

You can install it as a new operating system, or upgrade to it from an Ubuntu distribution.

http://environmentalomics.org/bio-linux/ (http://environmentalomics.org/bio-linux/)


# What if I don't want to install a new operating system?

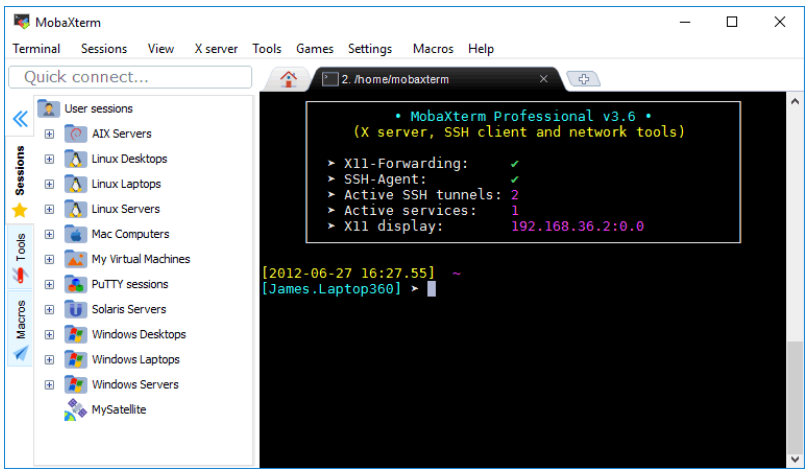
There are options to access and use a command line even without having to install a new operating system in your computer. Let's see some of them.

## Windows

MS Windows doesn't have a terminal emulator (except the latest versions), but there are a few options available.

- **Putty** (http://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html (http://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html)) is the simplest and most lightweight option to connect to a remote Unix server.

- **Putty Manager** (http://puttymanager.sourceforge.net/ (http://puttymanager.sourceforge.net/)) extends putty and provides multiple tabs and options.
- **Cygwin** (https://www.cygwin.com/ (https://www.cygwin.com/)) emulates a whole unix environment. You can execute commands on your computer, or connect to a remote server
- **MobaXTerm** (http://mobaxterm.mobatek.net/ (http://mobaxterm.mobatek.net/)) is a commercial solution (but the free version is usually enough) providing a Unix environment and some more options.



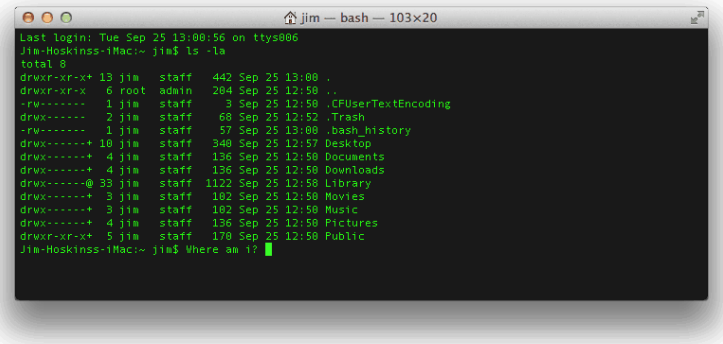
(http://mobaxterm.mobatek.net/)

To install new software in Cygwin or MobaXterm:, use (e.g. make):

```
apt-cyg install make
```

# Mac

MacOs is also a Unix system, and it comes with a teminal emulator already installed. You should be able to use the Console App in Mac.




# Linux

Congratulations on having Linux installed! There are several applications that you can use, from gnome-terminal to konsole.


# Time for a break!