Introduction to the command line (for humanists)

Moving around

pwd (print working directory) **Where am I** right now?
cd (**change directory**) Move to a different folder / directory
cd ../ Move **up** one directory
cd ../../ Move **up** two directories

QUICK TIP: use the **tab** key to auto complete.

The file system

/ (forward slash) **Root**, the top level of the filesystem
/Users/myname or **/home/myname** The **home** directory. Put your files here.
/Users/myname/Desktop or **/home/myname/desktop** The **desktop**
~ **Shortcut** code for **home** directory.

Seeing what you have

ls **List** the files in a directory
ls -a **List all** - show everything, do not hide dot files

ls -l (long list format) Show all the **details**
ls -al (long list format and all files) Gimme **everything**. I can handle it.
ls –hl (long list with human readable **filesizes**)
ls a* List all files that **start with** the letter 'a' (note: this goes into sub-folders. ls –d a* does not)
ls *.pdf List all the files with a .pdf **file extension**.
Other commands to try: ls -m, ls -b

Switch: a switch is an extra parameter added to a command. It is usually a hyphen followed by one or more letters.

Doing things

mkdir newdir (make directory) **Create** a new **directory**.
rmdir directoryname (remove directory) **Delete** a **directory**.

cp fromfile tofile (copy) Make a **copy** of 'fromfile' and name it 'tofile'
mv (move) **Move a file or directory.** (Be careful!)
mv myfile ../ **Move** 'myfile' **up** one directory.
mv myfile mydir/ **Move** 'myfile' down **into** the 'mydir' **directory**.
mv myfile.txt mynewfilename.txt **Rename** a **file**. Same as moving a file to a new name.
rm filename.txt **Remove** a **file**
executing applications
such as.... text editors: **nano**, **pico**, **vim**
nano myfile.txt **Open and edit** myfile.txt

Playing with text

grep "king" *.* **Search** for the **text** king in **any** file in the current directory.
grep "\bking\b" **Search** for the **word** 'king' in any file in the current directory
grep -n "\bking\b" Same, but **include line numbers.**
grep -r 'searchtext' **Search recursively** (from current directory downwards)

Note: The results of an operation, such as grep above, can be sent to a text file rather than the terminal by following the command with a greater than sign and then the name of the file to be created. For example:
grep "castle" *.* > output.txt
wc -w textfile.txt (word count) Count the number of words in a file.
wc –w *.txt

The second example uses what are called **'regular expressions'**. In this case, the code '\b' means 'word boundary, so '\bking\b' means the word 'king'.

sort myfile.txt **Sort** the lines in a file alphabetically.
uniq -c myfile.txt Remove duplicate lines from a file. (Useful for cleaning data.) Note: the 'c' switch here includes the number of occurences.

Finding things

find **Finds** everything in current and all sub-directories
find ./ **Finds** everything in current directory.
find / -name 'myfile.txt' From the root down, **find** all files **named** 'myfile.txt'
find / -name 'M*' From the root down, **find** all files that **start with** M

CTRL+C: To stop a command that's out of control, enter Ctrl+C. This will 'kill' it (cill it?).

Permissions and ownership

If you run ls -l on a directory, you will see, in the first column, something like '-rw-rw-r--' or 'drwxrwxr--'. This is a string of characters that represents the permissions on each file or directory. The presence of a 'd' at the beginning means it is a directory. The rest are three sets of three characters, 'r','w',and 'x'. These stand for **read**, **write**, and **execute**. The first group id for the **user** who owns the file, the second is for the **group** of users associated with the file, and the third is for **other**, the whole world. So, -rwxr-xr—translates to a file (since there is no 'd'), 'rwx', meaning the owner can read, write, and execute this file, 'r-x' meaning the group can read and execute, but not write to this file, and finally 'r--', meaning the rest of the world can only read the file and not write to it. Let's take a break.

To change permissions:
chmod is used to change permissions. The command is followed by u,g,or o (user, group, other), a plus or a minus sign and r,w, or x (read, write, or execute) For example:
chmod u+x myfile.txt Give the **owner** (user) permission to **execute** the file.
chmod g-w myfile.txt Remove write permissions for the group.

superdo

If permissions prevent you from doing something you want to do, most likely you should not be doing what you are trying to do. But if you must, and your account on the computer has administrative privileges, you can override permission problems by temporarily requesting administrative privileges. A super doer is a user who is able to masquerade as the administrator. The command sudo, prepended to another command, attempts to run that command as the administrator (usually called root):
sudo chmod o+rwx myfile.txt Enable read, write, and execute permissions on myfile.txt as root

If your account is able to do this, please be careful.