Shell

The terminal is the "window" (more or less), while the shell is a program (or a programming language, like R and Python are). The shell has been around longer than most of its users have been alive. It has survived because it's a powerful tool that allows users to perform complex and powerful tasks, often with just a few keystrokes or lines of code. It helps users automate repetitive tasks and easily combine smaller tasks into larger, more powerful workflows. There are several shell programs, `bash` (and `zsh`) being the most common. They are almost equivalent. For Window users, `PowerShell` is the command shell and scripting language.

Summary

- Shell References: 2. navigating files and directories and 3. working with files and directories from software carpentry introduction.
- Directory structure, root is `/`
- Shortcuts:
  - `.` : the current directory
  - `..` : goes up one level from the current directory
  - `~` : the current user's home directory
  - `-` : previous directory I was in
- Tab completion to get program and file names
- Up/down arrows and `!` to repeat command
- Note that mostly commands which are used in Bash can be used in PowerShell like `'rm'`, `'ls'`, `'cp'`, but it is not always the case. See some examples.

Bash Command	Task	Quiz
`whoami`	who am I? to get your username	O
`echo`	print	O
`pwd`	print working directory. where am I?	O
`ls`	list	O
`cd`	change directory	O
`mkdir`	make directory	O
`rm`	remove (forever)	O
`rmdir`	remove (delete) directory, if empty	O
`mv`	move (and rename). can overwrite existing files, unless `-i` to ask	O
`cp`	copy. would also overwrite existing files	O
`touch`	create blank file, or modify time stamp of existing file	
`nano`	run a text editor called Nano to create/modify a file	
`diff`	difference	
`wc`	word count: lines, words, characters. `-l`, `-w`, `-c`	
`cat`	concatenate	

Bash Command	Task	Quiz
`less`	because "less is more". `q` to quit.	
`sort`	`-n` for numerical sorting.	
`head`	first 10 lines. `-n 3` for first 3 lines (etc.)	
`tail`	last 10 lines. `-n 3` for last 3 lines, `-n +30` for line 30 and up	
`uniq`	filters out repeated lines (consecutive). `-c` to get counts	
`cut`	cut and return column(s). `-d`, to set the comma as field delimiter (tab otherwise), `-f2` to get 2nd field (column)	
`history`	shows the history of all previous commands, numbered	
`!`	`!76` to re-execute command number 76 in the history, !$ for last word or last command	

File names

So important: **no spaces!** example:

- create a directory 'data science' in `repos`, using a graphical user interface (GUI; e.g., Finder)
- try to remove it from the command line:

```
cd repos
ls
rmdir data science
```

How can we remove this directory?

- use `rmdir 'data science'` or `rmdir "data science"`.

Useful suggestions on file names

- prefer lower-case letters, especially for the first letter of a file name: time saver, along with tab completion
- common usage: capitalize between words, or underscores, or `-`, like `shellWarmupActivities` (camel case style) or `shell_warmup_activities` (snake case style).
- use ASCII characters only, no space, no `/`, no `\` (for Windows), no `-` for the first character.
- R users: avoid dots. conventionally used for the file extension.