



DIS08 – Data Modeling

02 – Working with the Unix Shell

Philipp Schaer, Technische Hochschule Köln, Cologne, Germany

Version: WS 2021

Disclaimer

This lesson is based on the Library Carpentry

- <https://librarycarpentry.org/lc-shell/>
- Library Carpentry develops lessons and teaches workshops for and with people working in library- and information-related roles.
- Our goal is to create an on-ramp to empower this community to use software and data in their own work as well as be advocates for and train others in efficient, effective and reproducible data and software practices.



Agenda

This week's agenda

- What is the shell?
- Files and directories
- History and tab completion
- Counting and sorting contents in files
- Pipes and redirection

Next week

- Version Control Systems
- Git
- GitHub

The image shows a computer screen with two windows. The terminal window displays the results of an nmap scan on host 10.2.2.2, showing open ports 80 and 81. It also shows the start of an sshnuke exploit attempt, with assembly code for the RCR, BSR, SHRD, and CLD instructions. The exploit editor window shows the assembly code:

```
EDIT01 sshnuke
rcr ebx, 1
bsr ecx, ecx
shrd ebx, edi, cl
clt d ax, adv, cl
[nobile]
```

Below the terminal window, a large text overlay reads: "Starting nmap 0.2.54BETA25 Insufficient responses for TCP sequencing (3), OS detection may be less accurate (The 1539 ports scanned but not shown below are in state: closed)".

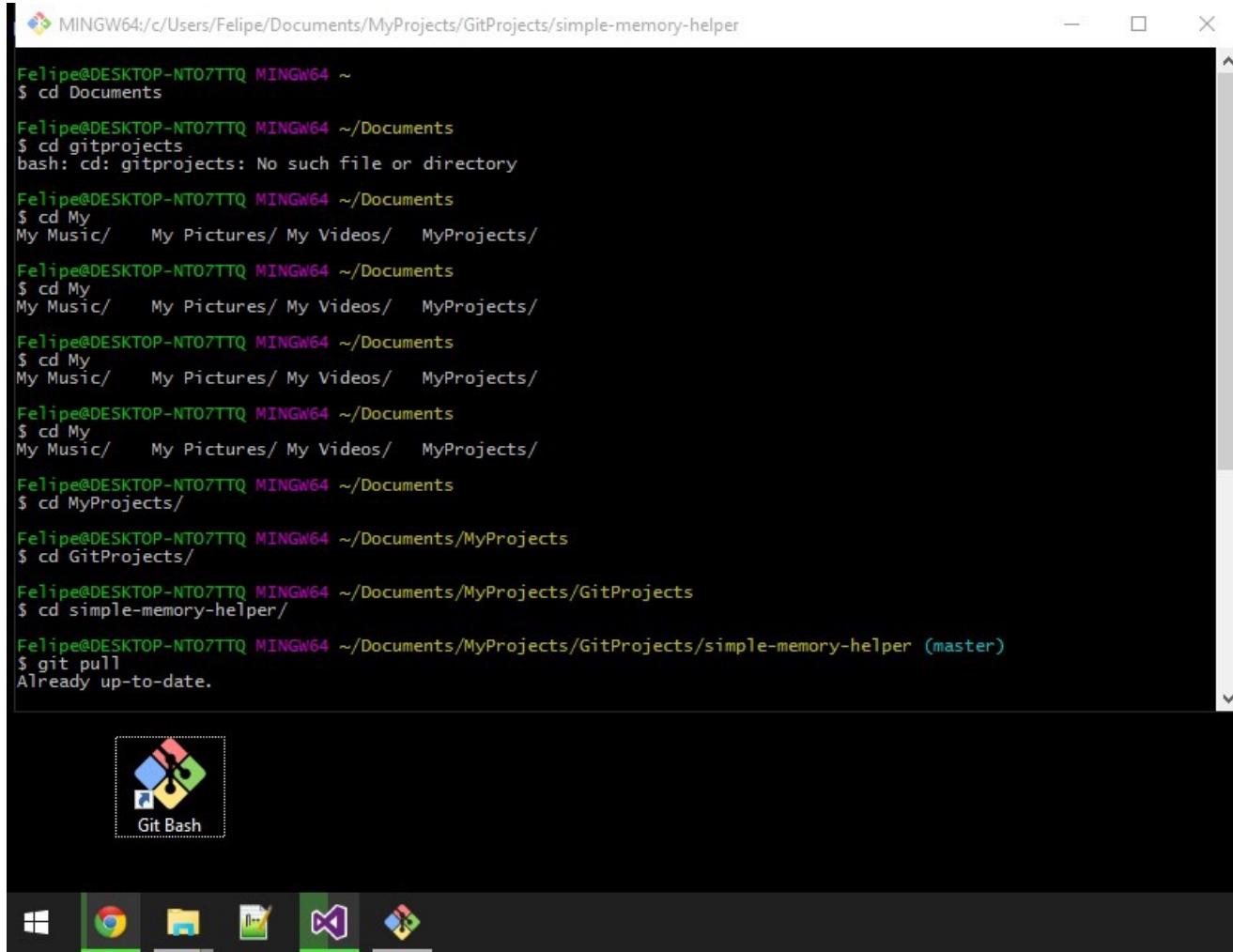
What is the shell?

And why should we use it?

Terminal on the Mac

```
pschaer@linux2: /dat... #1      bash      #2
drwxr-xr-x  22 schaer staff   704B Apr 15 17:42 _includes/
drwxr-xr-x   9 schaer staff   288B Apr 15 17:42 _layouts/
-rw-r--r--   1 schaer staff   1.2K Apr 15 17:42 aio.md
drwxr-xr-x   7 schaer staff  224B Apr 15 17:42 assets/
drwxr-xr-x  16 schaer staff   512B Apr 15 17:42 bin/
drwxr-xr-x   3 schaer staff   96B Apr 15 17:42 code/
-rw-r--r--   1 schaer staff  745B Apr 15 17:42 contribute.md
drwxr-xr-x  12 schaer staff  384B Apr 15 17:42 data/
drwxr-xr-x   6 schaer staff  192B Apr 15 17:42 fig/
drwxr-xr-x   3 schaer staff   96B Apr 15 17:42 files/
-rw-r--r--   1 schaer staff  817B Apr 15 17:42 index.md
-rw-r--r--   1 schaer staff  2.5K Apr 15 17:42 reference.md
-rw-r--r--   1 schaer staff  2.6K Apr 15 17:42 setup.md
schaer@touchbot:~/sciebo/00-LEHRE/DIS08-Data Modeling/03-shell$ ls
-bash: l: command not found
schaer@touchbot:~/sciebo/00-LEHRE/DIS08-Data Modeling/03-shell$ ls
AUTHORS          README.md        _layouts       data
CITATION         _config.yml     aio.md        fig
CODE_OF_CONDUCT.md _episodes      assets        files
CONTRIBUTING.md  _episodes_rmd   bin           index.md
LICENSE.md       _extras         code          reference.md
Makefile          _includes      contribute.md  setup.md
schaer@touchbot:~/sciebo/00-LEHRE/DIS08-Data Modeling/03-shell$ ls | wc
      24      24     228
schaer@touchbot:~/sciebo/00-LEHRE/DIS08-Data Modeling/03-shell$
```

Git Bash on Windows

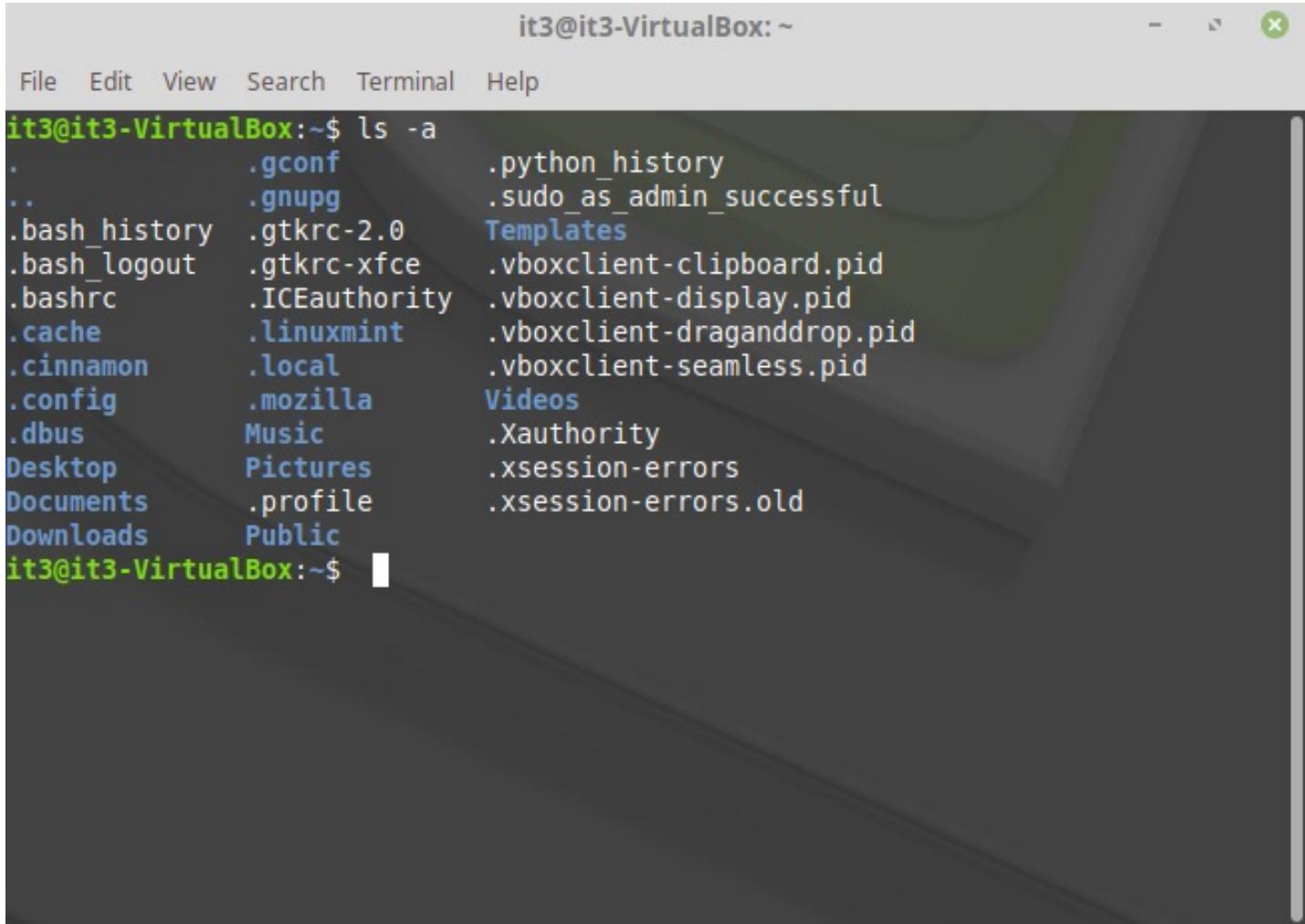


The screenshot shows a Windows desktop environment with a Git Bash terminal window open. The terminal window has a title bar 'MINGW64:/c/Users/Felipe/Documents/MyProjects/GitProjects/simple-memory-helper'. The command history in the terminal is as follows:

```
Felipe@DESKTOP-NT07TTQ MINGW64 ~
$ cd Documents
Felipe@DESKTOP-NT07TTQ MINGW64 ~/Documents
$ cd gitprojects
bash: cd: gitprojects: No such file or directory
Felipe@DESKTOP-NT07TTQ MINGW64 ~/Documents
$ cd My
My Music/ My Pictures/ My Videos/ MyProjects/
Felipe@DESKTOP-NT07TTQ MINGW64 ~/Documents
$ cd My
My Music/ My Pictures/ My Videos/ MyProjects/
Felipe@DESKTOP-NT07TTQ MINGW64 ~/Documents
$ cd My
My Music/ My Pictures/ My Videos/ MyProjects/
Felipe@DESKTOP-NT07TTQ MINGW64 ~/Documents
$ cd My
My Music/ My Pictures/ My Videos/ MyProjects/
Felipe@DESKTOP-NT07TTQ MINGW64 ~/Documents
$ cd MyProjects/
Felipe@DESKTOP-NT07TTQ MINGW64 ~/Documents/MyProjects
$ cd GitProjects/
Felipe@DESKTOP-NT07TTQ MINGW64 ~/Documents/MyProjects/GitProjects
$ cd simple-memory-helper/
Felipe@DESKTOP-NT07TTQ MINGW64 ~/Documents/MyProjects/GitProjects/simple-memory-helper (master)
$ git pull
Already up-to-date.
```

Below the terminal window, the Windows taskbar is visible with several pinned icons: File Explorer, Microsoft Edge, File Explorer again, Microsoft Word, Microsoft Visual Studio, and File Explorer again.

Terminal on Linux



A screenshot of a Linux terminal window titled "it3@it3-VirtualBox: ~". The window has a standard title bar with icons for minimize, maximize, and close. Below the title bar is a menu bar with "File", "Edit", "View", "Search", "Terminal", and "Help". The main area of the terminal shows the output of the command "ls -a", which lists all files and directories in the current directory (~). The output includes hidden files like ".gconf", ".gnupg", and ".profile", as well as other files like ".bash_history", ".bash_logout", ".bashrc", ".cache", ".cinnamon", ".config", ".dbus", ".Desktop", ".Documents", ".Downloads", ".gtkrc-2.0", ".gtkrc-xfce", ".ICEauthority", ".linuxmint", ".local", ".mozilla", "Music", "Pictures", "Public", ".pytho_n_history", ".sudo_as_admin_successful", "Templates", ".vboxclient-clipboard.pid", ".vboxclient-display.pid", ".vboxclient-draganddrop.pid", ".vboxclient-seamless.pid", "Videos", ".Xauthority", ".xsession-errors", and ".xsession-errors.old". The terminal prompt "it3@it3-VirtualBox: ~" is visible at the bottom, followed by a cursor icon.

```
it3@it3-VirtualBox:~$ ls -a
.
..
.bash_history .gconf      .python_history
..             .gnupg      .sudo_as_admin_successful
.bash_logout   .gtkrc-2.0  Templates
.bashrc        .gtkrc-xfce .vboxclient-clipboard.pid
.cache         .ICEauthority .vboxclient-display.pid
.cinnamon     .local       .vboxclient-draganddrop.pid
.config        .mozilla    .vboxclient-seamless.pid
.dbus          Music       .Xauthority
.Desktop      Pictures    .xsession-errors
/Documents    .profile    .xsession-errors.old
.Downloads    Public
it3@it3-VirtualBox:~$
```

What is the shell, and why to use it? #1

- The shell (sometimes referred to as the "Unix shell") is a program that allows you to **interact with your computer** using **typed text commands**.
- It is the primary interface used on **Linux** and **Unix-based** systems, such as **MacOS**, and can be installed optionally on other operating systems such as **Windows**.
- It is a "**command line interface**"
 - **instructions are given to the computer** by typing in commands, and it responds by performing a task or generating an output.
 - This output is often **directed to the screen**, but can also be directed to a file, or even to other commands, creating powerful chains of actions with very little effort.

What is the shell, and why to use it? #2

- Using a shell sometimes feels more like programming than like using a mouse.
- Commands are “**complicated**” and **terse**, their names are frequently **cryptic**, and their output is lines of text rather than something visual like a graph.
- On the other hand, with only a few keystrokes, the shell allows you to **combine existing tools into powerful pipelines** and to handle large volumes of data automatically.
- This automation not only **makes you more productive**, but also improves the **reproducibility** of your workflows by allowing you to save and then repeat them with a few simple commands.
- **Understanding the basics of the shell** provides a **useful foundation** for learning to program, since some of the concepts you will learn here will translate to programming.

What is the shell, and why to use it? #3

- The shell is one of the **most productive programming environments ever created**. Once mastered, you can use it to experiment with different commands interactively, then use what you have learned to automate your work.
- We will introduce **task automation** by looking at how data can be manipulated, counted, and mined using the shell.
- We will cover a small number of **basic commands**, which will constitute **building blocks** upon which more complex commands can be constructed to fit your data or project.
- Even if you do not do your own programming or your work currently does not involve the command line, **knowing some basics about the shell can be useful**.

Power of the Shell – An example

I got a zipped data set with ~50 Gbyte in a myriad of files.

- How many files are there?
 - \$ ls | wc
- How many files are in XML format?
 - \$ ls *.xml | wc
- Move only a subset (XML files) to another folder
 - \$ mkdir ../xml
 - \$ mv *.xml ../xml
- All this is **impossible** using the Finder or the Explorer!

3. bash

schaer@touchbot:~\$

Command line

\$ = Prompt shows that the command line
is ready for input.

3. bash

```
schaer@touchbot:~$ pwd  
/Users/schaer  
schaer@touchbot:~$
```

pwd is a command that shows the path you are in.

Command! After you hit enter the command is executed.

After the result was printed the command line is ready again.

3. bash

```
schaer@touchbot:~$ pwd
/Users/schaer
schaer@touchbot:~$ ls
Applications           OneDrive          centre-clef19
Desktop                Pictures          eBooks
Documents              Public            ir.web.th-koeln.de
Downloads              PycharmProjects sciebo
Dropbox                Research          solr
Games                  Sites             src
Google Drive           VirtualBox VMs  stella
Library                Zotero            temp
Movies                 __MACOSX        workspace
Music                  archive
0IR2018-SpecialIssueBias bin
schaer@touchbot:~$
```

ls lists the content of a path/folder.

```
schaer@touchbot:~$ pwd  
/Users/schaer  
schaer@touchbot:~$ ls  
Applications OneDrive centre-clef19  
Desktop Pictures eBooks  
Documents Public ir.web.th-koeln.de  
Downloads PycharmProjects sciebo  
Dropbox Research solr  
Games Sites src  
Google Drive VirtualBox VMs stella  
Library Zotero temp  
Movies __MACOSX workspace  
Music archive  
OIR2018-SpecialIssueBias bin  
schaer@touchbot:~$ ls -l
```

Commands can take parameters, options or arguments

Arguments modify the workings of the command by telling the computer what sort of output or manipulation we want.

3. bash

```

drwx-----+ 10 schaer staff    320 Oct 14 2018 Movies
drwx-----+  4 schaer staff    128 Sep 13 2017 Music
drwxr-xr-x 18 schaer staff    576 Oct 17 17:00 OIR2018-SpecialIssueBias
drwx-----@ 13 schaer staff    416 Apr  8 21:38 OneDrive
drwx-----+ 26 schaer staff    832 Aug 17 2017 Pictures
drwxr-xr-x+  5 schaer staff    160 Aug  9 2017 Public
drwxr-xr-x  5 schaer staff    160 Apr 10 2018 PycharmProjects
drwxr-xr-x 10 schaer staff    320 Apr 10 2017 Research
drwxr-xr-x 26 schaer staff    832 Jan 11 14:26 Sites
drwxr-xr-x  6 schaer staff    192 Mar 30 22:50 VirtualBox VMs
drwxr-xr-x 17 schaer staff    544 Apr 16 08:20 Zotero
drwxr-xr-x  4 schaer staff    128 Nov 13 2017 __MACOSX
drwxr-xr-x  8 schaer staff    256 Apr 16 18:26 archive
drwxr-xr-x 10 schaer staff    320 Dec  7 2017 bin
drwxr-xr-x 13 schaer staff    416 Apr 16 18:25 centre-clef19
drwxr-xr-x 77 schaer staff   2464 Jan  9 2018 eBooks
lrw.....x...  .w.....x... drwxr-xr-x 17 schaer staff    544 Dec 26 23:22 src
drwxr-xr-x  1 schaer staff     20 Jan 11 14:27 stella -> Sites/stella-website
drwxr-xr-x  8 schaer staff    256 Feb 22 12:51 temp
drwxr-xr-x 18 schaer staff    576 Jan  8 2016 workspace
schaer@touchbot:~$ 

```

ls -l gives us a LONG version of the output, so we see dates, file sizes, file owners, etc.

3. bash

```

drwx-----+ 10 schaer staff 320B Oct 14 2018 Movies
drwx-----+ 4 schaer staff 128B Sep 13 2017 Music
drwxr-xr-x 18 schaer staff 576B Oct 17 17:00 OIR2018-SpecialIssueBias
drwx-----@ 13 schaer staff 416B Apr 8 21:38 OneDrive
drwx-----+ 26 schaer staff 832B Aug 17 2017 Pictures
drwxr-xr-x+ 5 schaer staff 160B Aug 9 2017 Public
drwxr-xr-x 5 schaer staff 160B Apr 10 2018 PycharmProjects
drwxr-xr-x 10 schaer staff 320B Apr 10 2017 Research
drwxr-xr-x 26 schaer staff 832B Jan 11 14:26 Sites
drwxr-xr-x 6 schaer staff 192B Mar 30 22:50 VirtualBox VMs
drwxr-xr-x 17 schaer staff 544B Apr 16 08:20 Zotero
drwxr-xr-x 4 schaer staff 128B Nov 13 2017 __MACOSX
drwxr-xr-x 8 schaer staff 256B Apr 16 18:26 archive
drwxr-xr-x 10 schaer staff 320B Dec 7 2017 bin
drwxr-xr-x 13 schaer staff 416B Apr 16 18:25 centre-clef19
drwxr-xr-x 77 schaer staff 2.4K Jan 9 2018 eBooks

lrwrxr-xr-x 1 schaer staff 20B Jan 11 14:27 stella -> Sites/stella-website
drwxr-xr-x 8 schaer staff 256B Feb 22 12:51 temp
drwxr-xr-x 18 schaer staff 576B Jan 8 2016 workspace
schaer@touchbot:~$ 
```

Sites/ir

Same command but an additional argument: ls -lh
 It gives us a human-readable version of the output,
 with more clear file sizes.

Y

```

4. sh

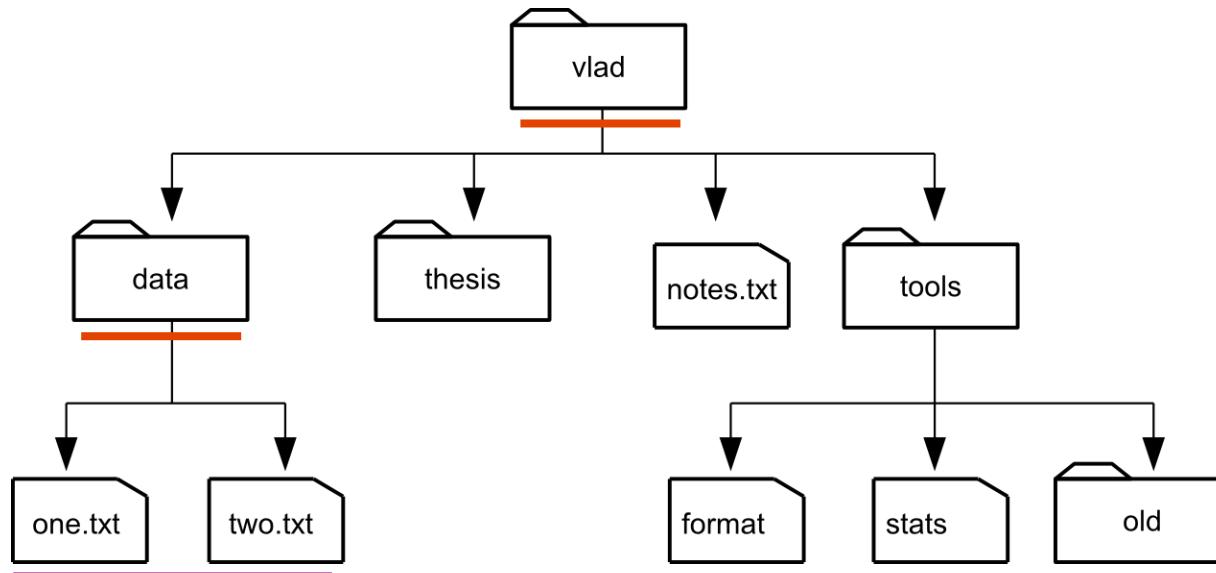
drwx-----@ 13 schaer staff 416B Apr  8 21:38 OneDrive
drwx-----+ 26 schaer staff 832B Aug 17 2017 Pictures
drwxr-xr-x+  5 schaer staff 160B Aug  9 2017 Public
drwxr-xr-x   5 schaer staff 160B Apr 10 2018 PycharmProjects
drwxr-xr-x   10 schaer staff 320B Apr 10 2017 Research
drwxr-xr-x  26 schaer staff 832B Jan 11 14:26 Sites
drwxr-xr-x   6 schaer staff 192B Mar 30 22:50 VirtualBox VMs
drwxr-xr-x   17 schaer staff 544B Apr 16 08:20 Zotero
drwxr-xr-x   4 schaer staff 128B Nov 13 2017 __MACOSX
drwxr-xr-x   8 schaer staff 256B Apr 16 18:26 archive
drwxr-xr-x   10 schaer staff 320B Dec  7 2017 bin
drwxr-xr-x   13 schaer staff 416B Apr 16 18:25 centre-clef19
drwxr-xr-x   77 schaer staff 2.4K Jan  9 2018 eBooks
lrvwxr-xr-x   1 schaer staff 24B Aug 17 2017 ir.web.th-koeln.de -> Sites/ir
.web.th-koeln.de
drwx-----  26 schaer staff 832B Apr 16 18:13 sciebo
lrvwxr-xr-x   1 schaer staff 15B Nov 29 15:59 solr -> src/solr-7.5.0/
drwxr-xr-x   17 schaer staff 544B Dec 26 23:22 src
lrvwxr-xr-x   1 schaer staff 20B Jan 11 14:27 stella -> Sites/stella-website
drwxr-xr-x   8 schaer staff 256B Feb 22 12:51 temp
drwxr-xr-x   18 schaer staff 576B Jan  8 2016 workspace

schaer@touchbot:~$ cd src
schaer@touchbot:~/src$ pwd
/Users/schaer/src
schaer@touchbot:~/src$ 
```

We can change directories by using `cd`.

Paths in a filesystem

- Paths guide the way through the **filesystem** giving access to all **directories/folders** and **files**.
- The filesystem is structured as a tree with a clear hierarchy.



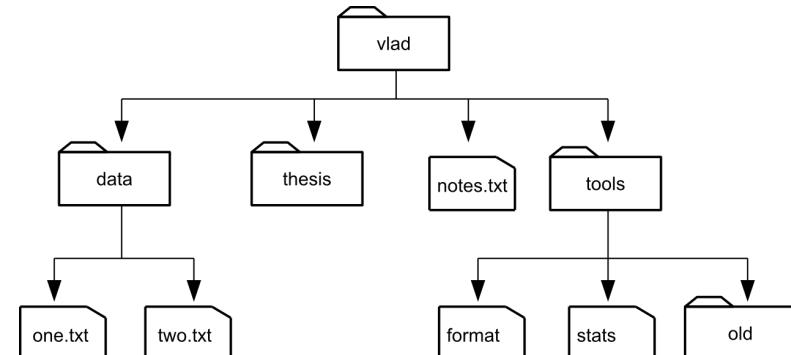
- A path to a file would look like this: /vlad/data/one.txt

Absolute vs. relative paths

Path can be **absolute** or **relative**.

- Absolute paths always show the complete path starting from the **root** of the filesystem (marked by /)
- Relative paths show the path from your **current position** in the filesystem

- Example: Current position is /vlad/data
- The following paths to one.txt are the same:
 - /vlad/data/one.txt
 - ./one.txt
 - ../data/one.txt



```

4. sh

drwxr-xr-x  5 schaer  staff   160B Apr 10 2018 PycharmProjects
drwxr-xr-x 10 schaer  staff   320B Apr 10 2017 Research
drwxr-xr-x 26 schaer  staff   832B Jan 11 14:26 Sites
drwxr-xr-x  6 schaer  staff   192B Mar 30 22:50 VirtualBox VMs
drwxr-xr-x 17 schaer  staff   544B Apr 16 08:20 Zotero
drwxr-xr-x  4 schaer  staff   128B Nov 13 2017 __MACOSX
drwxr-xr-x  8 schaer  staff   256B Apr 16 18:26 archive
drwxr-xr-x 10 schaer  staff   320B Dec  7 2017 bin
drwxr-xr-x 13 schaer  staff   416B Apr 16 18:25 centre-clef19
drwxr-xr-x 77 schaer  staff   2.4K Jan  9 2018 eBooks
lrwxr-xr-x  1 schaer  staff   24B Aug 17 2017 ir.web.th-koeln.de -> Sites/ir
.web.th-koeln.de
drwx----- 26 schaer  staff   832B Apr 16 18:13 sciebo
lrwxr-xr-x  1 schaer  staff   15B Nov 29 15:59 solr -> src/solr-7.5.0/
drwxr-xr-x 17 schaer  staff   544B Dec 26 23:22 src
lrwxr-xr-x  1 schaer  staff   20B Jan 11 14:27 stella -> Sites/stella-website
drwxr-xr-x  8 schaer  staff   256B Feb 22 12:51 temp
drwxr-xr-x 18 schaer  staff   576B Jan  8 2016 workspace

schaer@touchbot:~$ cd src
schaer@touchbot:~/src$ pwd
/Users/schaer/src
schaer@touchbot:~/src$ cd ..
schaer@touchbot:~$ pwd
/Users/schaer
schaer@touchbot:~$ 
```

cd .. tells the system to go back one level.

```

4. sh 🎙
drwxr-xr-x 17 schaer staff 544B Apr 16 08:20 Zotero
drwxr-xr-x 4 schaer staff 128B Nov 13 2017 __MACOSX
drwxr-xr-x 8 schaer staff 256B Apr 16 18:26 archive
drwxr-xr-x 10 schaer staff 320B Dec 7 2017 bin
drwxr-xr-x 13 schaer staff 416B Apr 16 18:25 centre-clef19
drwxr-xr-x 77 schaer staff 2.4K Jan 9 2018 eBooks
lrwxr-xr-x 1 schaer staff 24B Aug 17 2017 ir.web.th-koeln.de -> Sites/ir
.web.th-koeln.de
drwx----- 26 schaer staff 832B Apr 16 18:13 sciebo
lrwxr-xr-x 1 schaer staff 15B Nov 29 15:59 solr -> src/solr-7.5.0/
drwxr-xr-x 17 schaer staff 544B Dec 26 23:22 src
lrwxr-xr-x 1 schaer staff 20B Jan 11 14:27 stella -> Sites/stella-website
drwxr-xr-x 8 schaer staff 256B Feb 22 12:51 temp
drwxr-xr-x 18 schaer staff 576B Jan 8 2016 workspace

schaer@touchbot:~$ cd src
schaer@touchbot:~/src$ pwd
/Users/schaer/src
schaer@touchbot:~/src$ cd ..
schaer@touchbot:~$ pwd
/Users/schaer
schaer@touchbot:~$ cd .
schaer@touchbot:~$ pwd
/Users/schaer
schaer@touchbot:~$ cd ./src
schaer@touchbot:~/src$ █

```

While .. indicates to go up one level,
 . is an indicator for the current level!

```
4. sh  
drwx----- 26 schaer staff 832B Apr 16 18:13 sciebo  
lwxr-xr-x 1 schaer staff 15B Nov 29 15:59 solr -> src/solr-7.5.0/  
drwxr-xr-x 17 schaer staff 544B Dec 26 23:22 src  
lwxr-xr-x 1 schaer staff 20B Jan 11 14:27 stella -> Sites/stella-website  
drwxr-xr-x 8 schaer staff 256B Feb 22 12:51 temp  
drwxr-xr-x 18 schaer staff 576B Jan 8 2016 workspace  
schaer@touchbot:~$ cd src  
schaer@touchbot:~/src$ pwd  
/Users/schaer/src  
schaer@touchbot:~/src$ cd ..  
schaer@touchbot:~$ pwd  
/Users/schaer  
schaer@touchbot:~$ cd .  
schaer@touchbot:~$ pwd  
/Users/schaer  
schaer@touchbot:~$ cd ./src  
schaer@touchbot:~/src$ ls  
README          lemur-master      solr-7.0.1        trec_eval.9.0  
indri-5.8       lemur-master.zip  solr-7.1.0        trec_eval.9.0.tar  
indri-5.8.tar   elevation        solr-7.3.1  
lemur-4.12.tar  slack-export     solr-7.5.0  
schaer@touchbot:~/src$ cd ../src/solr-7.0.1  
schaer@touchbot:~/src/solr-7.0.1$ pwd  
/Users/schaer/src/solr-7.0.1  
schaer@touchbot:~/src/solr-7.0.1$
```

See how you can combine these levels!



4. sh

```
lrwxr-xr-x    1 schaer  staff   20B Jan 11 14:27 stella -> Sites/stella-website  
drwxr-xr-x    8 schaer  staff  256B Feb 22 12:51 temp  
drwxr-xr-x   18 schaer  staff  576B Jan  8 2016 workspace  
schaer@touchbot:~$ cd src  
schaer@touchbot:~/src$ pwd  
/Users/schaer/src  
schaer@touchbot:~/src$ cd ..  
schaer@touchbot:~$ pwd  
/Users/schaer  
schaer@touchbot:~$ cd .  
schaer@touchbot:~$ pwd  
/Users/schaer  
schaer@touchbot:~$ cd ./src  
schaer@touchbot:~/src$ ls  
README          lemur-master  
indri-5.8       lemur-master  
indri-5.8.tar    relevation      solr-7.3.1  
lemur-4.12.tar   slack-export     solr-7.5.0  
schaer@touchbot:~/src$ cd ../src/solr-7.0.1  
schaer@touchbot:~/src/solr-7.0.1$ pwd  
/Users/schaer/src/solr-7.0.1  
schaer@touchbot:~/src/solr-7.0.1$ cd ..  
schaer@touchbot:~/src$ mkdir "test folder"  
schaer@touchbot:~/src$ cd "test folder"  
schaer@touchbot:~/src/test folder$ █
```

Folders and filenames can include spaces
BUT it is not recommended!

If there are spaces in folders or filenames
you have to enclose the arguments in
quotation marks!

LS(1) BSD General Commands Manual LS(1)

NAME

ls -- list directory contents

SYNOPSIS

ls [-ABCDEFGHIJKLMNPQRSTUVWXYZ@abcdefghijklmnopqrstuvwxyzklmnopqrstuvwxyz1] [file ...]

DESCRIPTION

For each operand that names a file of a type other than directory, **ls** displays its name as well as any requested, associated information. For each operand that names a file of type directory, **ls** displays the names of files contained within that directory, as well as any requested, associated information.

If no operands are given, the contents of the current directory are displayed. If more than one operand is given, non-directory operands are displayed first; directory and non-directory operands are sorted separately and in lexicographical order.

The following options are available:

-@	Display extended information
:	Display the current working directory

You can get help on commands by using man, so man ls will give you a documentation for this command.

```
4. sh  
lrwxr-xr-x 1 schaer staff 20B Jan 11 14:27 stella -> Sites/stella-website  
drwxr-xr-x 8 schaer staff 256B Feb 22 12:51 temp  
drwxr-xr-x 18 schaer staff 576B Jan 8 2016 workspace  
schaer@touchbot:~$ cd src  
schaer@touchbot:~/src$ pwd  
/Users/schaer/src  
schaer@touchbot:~/src$ cd ..  
schaer@touchbot:~$ pwd
```

Take away message:

Knowing where you are in your directory structure is key to working with the shell!

```
schaer@touchbot:~/src$ cd ../src/solr-7.0.1  
schaer@touchbot:~/src/solr-7.0.1$ pwd  
/Users/schaer/src/solr-7.0.1  
schaer@touchbot:~/src/solr-7.0.1$ cd ..  
schaer@touchbot:~/src$ mkdir "test folder"  
schaer@touchbot:~/src$ cd "test folder"  
schaer@touchbot:~/src/test folder$ █
```

Some more shell “secrets”

Use TAB for auto-complete

- Hitting tab at any time within the shell will prompt it to attempt to auto-complete the line based on the files or sub-directories in the current directory.
- Where two or more files have the same characters, the auto-complete will only fill up to the first point of difference, after which we can add more characters, and try using tab again.

Command history

- Hit the Up key to see the last command, and the next, and the next
- `history` will print the whole history of all your last commands.

Canceling commands

- To cancel any ongoing processes in the Unix shell, hit Ctrl+X.

More shell commands

- cat read the content of a file and print it to stdout
- head read the beginning of a file and print it to stdout
- tail read the end of a file and print it to stdout
- less read the content of a file, print only so much that it fits on the screen. Hit spacebar to see next screen

These commands take many arguments like:

- head -n 20
- head file1 file2 file3

Any idea what this does?

Wildcards

- Wildcards are a feature of the shell and will therefore work with *any* command.
- The shell will expand wildcards to a list of files and/or directories before the command is executed, and the command will never see the wildcards.
 - * matches zero or more characters
 - ? matches exactly one character

```
ls *.pdf      -> ls 1.pdf temp.pdf klausur.pdf ...
```

```
ls c?de.pdf   -> ls cade.pdf code.pdf
```

Move, copy and delete files

- `mv` move / rename a file or folder
- `cp` copy a file or folder
- `rm` remove file
- `touch` create new (and empty) file
- `mkdir` create directories
- `rmdir` remove directories

All the details can be found by using [man](#)

Counting and mining with the shell

- We will begin by counting the contents of files using the Unix shell. We can use the Unix shell to quickly generate counts from across files, something that is tricky to achieve using the graphical user interfaces of standard office suites.
 - These commands are unlikely to revolutionise your work by themselves, but they're very versatile and will add to your foundation for working in the shell and for learning to code.
-
- `wc` number of lines, words, bytes and chars in files
 - `wc -l` number of lines in files
 - `sort` sort the content of a file, line by line
 - `sort -n` sort the content of a file, numerical-based

```
5. bash
schaer@touchbot:~/dis08/03-shell/shell-lesson$ ls
000003160_01_text.json    2014-01_JA.tsv          829-0.txt
2014-01-31_JA-africa.tsv  2014-02-02_JA-britain.tsv diary.html
2014-01-31_JA-america.tsv 33504-0.txt          lengths.txt
schaer@touchbot:~/dis08/03-shell/shell-lesson$ wc *.tsv
 13712 511261 3773660 2014-01-31_JA-africa.tsv
 27392 1049601 7731914 2014-01-31_JA-america.tsv
 507732 17606310 131122144 2014-01_JA.tsv
      5375 196999 1453418 2014-02-02_JA-britain.tsv
 554211 19364171 144081136 total
schaer@touchbot:~/dis08/03-shell/shell-lesson$ wc -l *.tsv
 13712 2014-01-31_JA-africa.tsv
 27392 2014-01-31_JA-america.tsv
 507732 2014-01_JA.tsv
      5375 2014-02-02_JA-britain.tsv
 554211 total
schaer@touchbot:~/dis08/03-shell/shell-lesson$
```

The output of wc is not sorted!
But we will get there...

Example: Finding the shortest file...

- Disclaimer: If we only have a handful of files to compare, it might be faster or more convenient to just check with Microsoft Excel, OpenRefine or your favourite text editor.
- BUT when we have tens, hundreds or thousands of documents, the Unix shell has a clear speed advantage.
- The real power of the shell comes from being able to **combine commands** and **automate tasks**, though. We will touch upon this slightly.
- For now, we'll see how we can build a simple pipeline to find the **shortest file** in terms of number of lines.

Pipes and redirections

- Up to now all output was written to `stdout` in the shell.
But you can *redirect* the output from the command to a file using the '**greater than**' sign (`>`):
 - `history > history.txt`
 - `cat *.txt > all.txt`
- An even mightier instrument is the **pipe** |
 - It allows to redirect the output to another program
 - `cat *.txt | wc`
- It can take some time to fully grasp pipes and use them efficiently, but it's a very powerful concept that you will find not only in the shell, but also in most programming languages.

```
5. bash
schaer@touchbot:~/dis08/03-shell/shell-lesson$ ls
000003160_01_text.json    2014-01_JA.tsv          829-0.txt
2014-01-31_JA-africa.tsv  2014-02-02_JA-britain.tsv diary.html
2014-01-31_JA-america.tsv 33504-0.txt          lengths.txt
schaer@touchbot:~/dis08/03-shell/shell-lesson$ wc *.tsv
 13712 511261 3773660 2014-01-31_JA-africa.tsv
 27392 1049601 7731914 2014-01-31_JA-america.tsv
 507732 17606310 131122144 2014-01_JA.tsv
      5375 196999 1453418 2014-02-02_JA-britain.tsv
 554211 19364171 144081136 total
schaer@touchbot:~/dis08/03-shell/shell-lesson$ wc -l *.tsv
 13712 2014-01-31_JA-africa.tsv
 27392 2014-01-31_JA-america.tsv
 507732 2014-01_JA.tsv
      5375 2014-02-02_JA-britain.tsv
 554211 total
schaer@touchbot:~/dis08/03-shell/shell-lesson$ wc -l *.tsv > lengths.txt
schaer@touchbot:~/dis08/03-shell/shell-lesson$ cat lengths.txt
 13712 2014-01-31_JA-africa.tsv
 27392 2014-01-31_JA-america.tsv
 507732 2014-01_JA.tsv
      5375 2014-02-02_JA-britain.tsv
 554211 total
schaer@touchbot:~/dis08/03-shell/shell-lesson$
```

Now the information is stored
in a file lengths.txt

```
5. bash  
507732 17606310 131122144 2014-01_JA.tsv  
      5375 196999 1453418 2014-02-02_JA-britain.tsv  
554211 19364171 144081136 total  
schaer@touchbot:~/dis08/03-shell/shell-lesson$ wc -l *.tsv  
13712 2014-01-31_JA-africa.tsv  
27392 2014-01-31_JA-america.tsv  
507732 2014-01_JA.tsv  
      5375 2014-02-02_JA-britain.tsv  
554211 total  
schaer@touchbot:~/dis08/03-shell/shell-lesson$ wc -l *.tsv > lengths.txt  
schaer@touchbot:~/dis08/03-shell/shell-lesson$ cat lengths.txt  
13712 2014-01-31_JA-africa.tsv  
27392 2014-01-31_JA-america.tsv  
507732 2014-01_JA.tsv  
      5375 2014-02-02_JA-britain.tsv  
554211 total  
schaer@touchbot:~/dis08/03-shell/shell-lesson$ sort -n lengths.txt > sorted-leng  
ths.txt  
schaer@touchbot:~/dis08/03-shell/shell-lesson$ cat sorted-lengths.txt  
      5375 2014-02-02_JA-britain.tsv  
13712 2014-01-31_JA-africa.tsv  
27392 2014-01-31_JA-america.tsv  
507732 2014-01_JA.tsv  
554211 total  
schaer@touchbot:~/dis08/03-shell/shell-lesson$
```

We are getting there...
How do we get just the first line?

```
5. bash  
554211 19364171 144081136 total  
schaer@touchbot:~/dis08/03-shell/shell-lesson$ wc -l *.tsv  
13712 2014-01-31_JA-africa.tsv  
27392 2014-01-31_JA-america.tsv  
507732 2014-01_JA.tsv  
5375 2014-02-02_JA-britain.tsv  
554211 total  
schaer@touchbot:~/dis08/03-shell/shell-lesson$ wc -l *.tsv > lengths.txt  
schaer@touchbot:~/dis08/03-shell/shell-lesson$ cat lengths.txt  
13712 2014-01-31_JA-africa.tsv  
27392 2014-01-31_JA-america.tsv  
507732 2014-01_JA.tsv  
5375 2014-02-02_JA-britain.tsv  
554211 total  
schaer@touchbot:~/dis08/03-shell/shell-lesson$ sort -n < lengths.txt > sorted-lengths.txt  
schaer@touchbot:~/dis08/03-shell/shell-lesson$ cat sorted-lengths.txt  
5375 2014-02-02_JA-britain.tsv  
13712 2014-01-31_JA-africa.tsv  
27392 2014-01-31_JA-america.tsv  
507732 2014-01_JA.tsv  
554211 total  
schaer@touchbot:~/dis08/03-shell/shell-lesson$ head -n 1 sorted-lengths.txt  
5375 2014-02-02_JA-britain.tsv  
schaer@touchbot:~/dis08/03-shell/shell-lesson$
```

But these are three steps...
And we need to store something
in a file... There is a better way!

Tada!

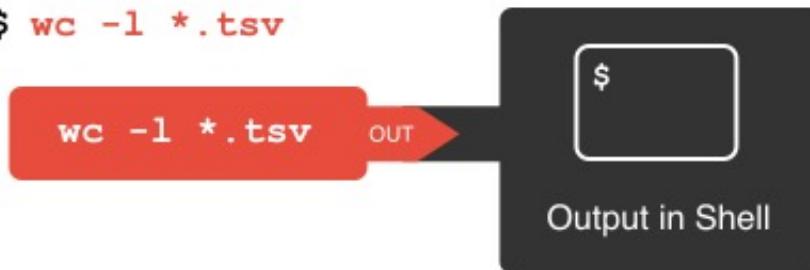
```
5. bash  
554211 total  
schaer@touchbot:~/dis08/03-shell/shell-lesson$ wc -l *.tsv > lengths.txt  
schaer@touchbot:~/dis08/03-shell/shell-lesson$ cat lengths.txt  
13712 2014-01-31_JA-africa.tsv  
27392 2014-01-31_JA-america.tsv  
507732 2014-01_JA.tsv  
5375 2014-02-02_JA-britain.tsv  
554211 total  
schaer@touchbot:~/dis08/03-shell/shell-lesson$ sort -n lengths.txt > sorted-lengths.txt  
schaer@touchbot:~/dis08/03-shell/shell-lesson$ cat sorted-lengths.txt  
5375 2014-02-02_JA-britain.tsv  
13712 2014-01-31_JA-africa.tsv  
27392 2014-01-31_JA-america.tsv  
507732 2014-01_JA.tsv  
554211 total  
schaer@touchbot:~/dis08/03-shell/shell-lesson$ head -n 1 sorted-lengths.txt  
5375 2014-02-02_JA-britain.tsv  
schaer@touchbot:~/dis08/03-shell/shell-lesson$ wc -l *.tsv | sort -n  
5375 2014-02-02_JA-britain.tsv  
13712 2014-01-31_JA-africa.tsv  
27392 2014-01-31_JA-america.tsv  
507732 2014-01_JA.tsv  
554211 total  
schaer@touchbot:~/dis08/03-shell/shell-lesson$
```

Same result as above!
Let's add another pipe...

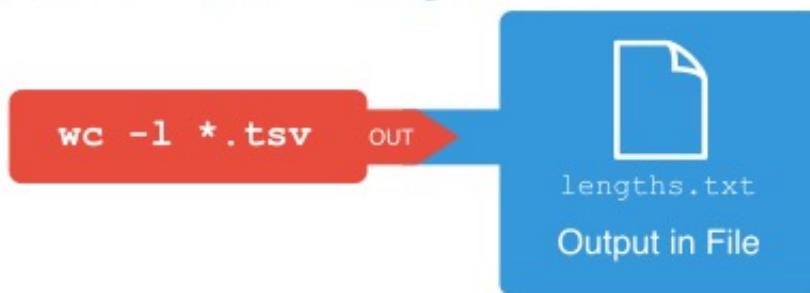
```
5. bash  
schaer@touchbot:~/dis08/03-shell/shell-lesson$ cat lengths.txt  
13712 2014-01-31 JA-africa.tsv  
27392 2014-  
507732 2014-  
5375 2014-  
554211 total  
schaer@touchbot:~/dis08/03-shell/shell-lesson$ sort -n lengths.txt > sorted-lengths.txt  
schaer@touchbot:~/dis08/03-shell/shell-lesson$ cat sorted-lengths.txt  
5375 2014-02-02_JA-britain.tsv  
13712 2014-01-31_JA-africa.tsv  
27392 2014-01-31_JA-america.tsv  
507732 2014-01_JA.tsv  
554211 total  
schaer@touchbot:~/dis08/03-shell/shell-lesson$ head -n 1 sorted-lengths.txt  
5375 2014-02-02_JA-britain.tsv  
schaer@touchbot:~/dis08/03-shell/shell-lesson$ wc -l *.tsv | sort -n  
5375 2014-02-02_JA-britain.tsv  
13712 2014-01-31_JA-africa.tsv  
27392 2014-01-31_JA-america.tsv  
507732 2014-01_JA.tsv  
554211 total  
schaer@touchbot:~/dis08/03-shell/shell-lesson$ wc -l *.tsv | sort -n | head -n 1  
5375 2014-02-02_JA-britain.tsv  
schaer@touchbot:~/dis08/03-shell/shell-lesson$
```

Solution in one line! And no need to store anything in a file anymore.

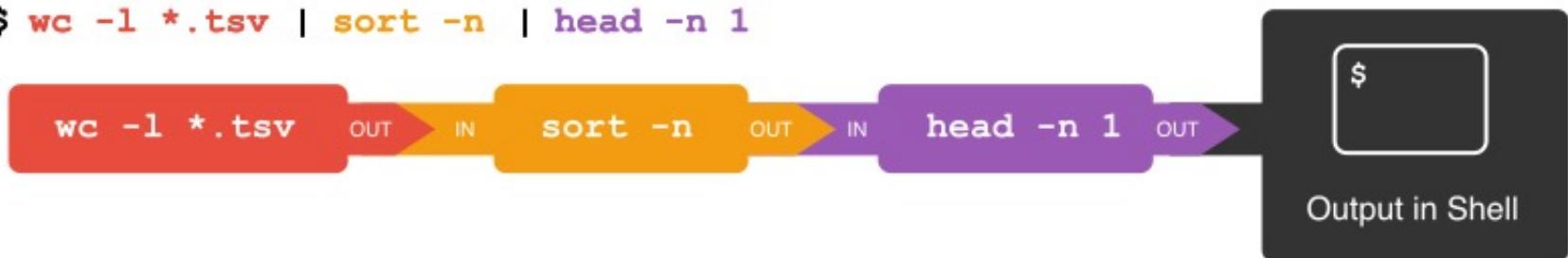
```
$ wc -l *.tsv
```



```
$ wc -l *.tsv > lengths.txt
```



```
$ wc -l *.tsv | sort -n | head -n 1
```



Pipes and filters

- This simple idea is why Unix has been so successful. Instead of creating enormous programs that try to do many different things, Unix programmers focus on creating **lots of simple tools that each do one job well**, and that **work well with each other**.
- This programming model is called “**pipes and filters**”.
- We’ve already seen pipes; a filter is a program like `wc` or `sort` that transforms a stream of input into a stream of output.
- Almost all of the standard Unix tools can work this way: unless told to do otherwise, they read from standard input, do something with what they’ve read, and write to standard output.

Pipes and filters

- The key is that any program that reads lines of text from standard input and writes lines of text to standard output can be combined with every other program that behaves this way as well. You can and should write your programs this way so that you and other people can put those programs into pipes to multiply their power.

Quiz time!

- `wc -l *.tsv | sort -n | head -n 1`
- We know what this pipeline does...!
- What would happen if you piped this into `cat`?

Exercise 2: The Unix Shell (3 pts)

2.1: Shell Tutorial

- Complete the lessons from the interactive online course at linuxjourney.com on learning the command line. Do the quizzes in each of the lectures.

2.2: Count the Lines

- Use your new knowledge to count the number of lines in your `introduction.md` file from exercise 1.
- Redirect the result to a file, e.g. `lines.txt`.
- Print the content of the file to the console.

More details on our central GitHub repo.