

Lecture 9: Bash Shell & Command Line

LING 1340/2340: Data Science for Linguists

Na-Rae Han

Objectives

- ▶ Finally, Bash shell
 - ◆ Running things in command line
 - ◆ Interacting with text files in command line
 - ◆ Regex-based text search using grep

Bash shell

▶ What is a "shell"?

- ◆ [https://en.wikipedia.org/wiki/Shell_\(computing\)](https://en.wikipedia.org/wiki/Shell_(computing))
- ◆ Usually refers to the command-line interface (CLI) as opposed to graphical user interface (GUI).
- ◆ **Bash** is the most common flavor of shell in Unix-like OS.

▶ Mac users

- ◆ Mac OS is a Unix-type OS.
- ◆ **Terminal** is a built-in shell, operates on Bash.

▶ Windows users

- ◆ We installed "**git bash**": a bash environment for running command-line git.
- ◆ As a bonus, it came with pretty much all of **popular Unix command-line tools!**

Resources

▶ Learning resources section:

- ◆ <https://naraehan.github.io/Data-Science-for-Linguists-2019/resources#bash>

▶ Software Carpentry, The Unix Shell:

- ◆ <http://swcarpentry.github.io/shell-novice/>

▶ Thirty Useful Unix Commands:

- ◆ <http://www.maths.manchester.ac.uk/~pjohnson/resources/unixShort/examples-commands.pdf>
 - ◆ You don't need: compress, finger, lpr, talk
 - ◆ (Windows) Use "less" instead of "more".

▶ grep and regular expressions:

- ◆ <https://www.regular-expressions.info/tutorial.html> (learn regex)
- ◆ <http://www.softpanorama.org/Tools/grep.shtml#Introduction> (Old-school site, but outstanding intro)

Shell introduction, navigating

- ▶ Introducing the shell
 - ◆ <http://swcarpentry.github.io/shell-novice/01-intro/>
- ▶ Navigating & working with files and directories
 - ◆ <http://swcarpentry.github.io/shell-novice/02-filedir/>
 - ◆ <http://swcarpentry.github.io/shell-novice/03-create/>
- ▶ We've been doing some of these already, as part of our git routine. You should know:
 - ◆ `.` `..` `~`
 - ◆ `pwd`
 - ◆ `cd`
 - ◆ `ls`
 - ◆ Command-line history with `↑` and `↓`
 - ◆ Using TAB for file name completion
 - ◆ Using Control+C to quit

Settling in, customizing

- ▶ You can customize your shell.

- ◆ `.bashrc`
- ◆ `.bash_profile`

← These files store your customization.

- ▶ In your **home directory**:

- ◆ `your_editor .bash_profile &`

- ◆ After adding entries or editing, you should either log back in, or execute `source .bash_profile`.

- ▶ Aliasing is the most common customization method:

```
alias calc='/c/windows/system32/calc.exe'  
alias ls='ls -hF --color=tty' ←
```

← Your favorite shortcuts and command-line options

Mac users: color option is not supported by default unless you customize Terminal.

PATH, which, where

- ▶ We have been occasionally using `pip` to install Python libraries. Where is this `pip`? Which pip are you using?

```
MINGW64:/c/Users/narae
narae@T450s MINGW64 ~
$ which pip
/c/ProgramData/Anaconda3/Scripts/pip

narae@T450s MINGW64 ~
$ which pip3
/c/Program Files (x86)/Python35-32/Scripts/pip3

narae@T450s MINGW64 ~
$ which -a pip
/c/ProgramData/Anaconda3/Scripts/pip
/c/Program Files (x86)/Python35-32/Scripts/pip

narae@T450s MINGW64 ~
$ echo $PATH
/c/Users/narae/bin:/mingw64/bin:/usr/local/bin:/usr/bin:/bin:/mingw64/bin:/usr/bin:/c/Users/narae/bin:/c/WINDOWS/system32:/c/WINDOWS:/c/WINDOWS/System32/wbem:/c/WINDOWS/System32/WindowsPowerShell/v1.0:/c/ProgramData/Oracle/Java/javapath:/c/Program Files (x86)/PDFtk Server/bin:/c/Program Files (x86)/Windows Live/Shared:/c/Program Files (x86)/Skype/Phone:/c/ProgramData/Anaconda3:/c/ProgramData/Anaconda3/Scripts:/c/ProgramData/Anaconda3/Library/bin:/c/Program Files (x86)/Pandoc:/c/Program Files/Intel/WiFi/bin:/c/Program Files/Common Files/Intel/WirelessCommon:/c/Program Files (x86)/Windows Kits/8.1/Windows Performance Toolkit:/c/Program Files (x86)/Python35-32:/c/Program Files (x86)/Python35-32/Scripts:/c/Users/narae/AppData/Local/Microsoft/WindowsApps:/c/Program Files/Intel/WiFi/bin:/c/Program Files/Common Files/Intel/WirelessCommon:/c/Users/narae/AppData/Local/atom/bin:/usr/bin/vendor_perl:/usr/bin/core_perl
```

1st hit in PATH

PATH, which, where

If you want to install tweepy for this version of python, you can do:

- (1) `pip3 install tweepy`
- (2) `/c/Program\ Files.../Scripts/pip install tweepy`
- (3) `cd` into `/c/Program Files.../Scripts` directory and then `./pip install tweepy`

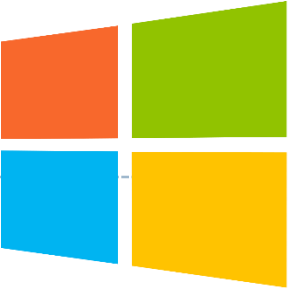
```
MINGW
narae@T450s MINGW64 ~
$ which pip
/c/ProgramData/Anaconda3/Scripts/pip

narae@T450s MINGW64 ~
$ which pip3
/c/Program Files (x86)/Python35-32/Scripts/pip3

narae@T450s MINGW64 ~
$ which -a pip
/c/ProgramData/Anaconda3/Scripts/pip
/c/Program Files (x86)/Python35-32/Scripts/pip ← 1st hit in PATH

narae@T450s MINGW64 ~
$ echo $PATH
/c/Users/narae/bin:/mingw64/bin:/usr/local/bin:/usr/bin:/bin:/mingw64/bin:/usr/b
in:/c/Users/narae/bin:/c/WINDOWS/system32:/c/WINDOWS:/c/WINDOWS/System32/wbem:/c
/WINDOWS/System32/WindowsPowerShell/v1.0:/c/ProgramData/Oracle/Java/javapath:/c/
Program Files (x86)/PDFtk Server/bin:/c/Program Files (x86)/windows Live/Shared:
/c/Program Files (x86)/Skype/Phone:/c/ProgramData/Anaconda3:/c/ProgramData/Anaco
nda3/Scripts:/c/ProgramData/Anaconda3/Library/bin:/c/Program Files (x86)/Pandoc:
/c/Program Files/Intel/WiFi/bin:/c/Program Files/Common Files/Intel/WirelessComm
on:/c/Program Files (x86)/Windows Kits/8.1/Windows Performance Toolkit:/c/Progra
m Files (x86)/Python35-32:/c/Program Files (x86)/Python35-32/Scripts:/c/Users/na
rae/AppData/Local/Microsoft/WindowsApps:/c/Program Files/Intel/WiFi/bin:/c/Progr
am Files/Common Files/Intel/WirelessCommon:/c/Users/narae/AppData/Local/atom/bin
:/usr/bin/vendor_perl:/usr/bin/core_perl
```


Windows users



- ▶ Because git-bash is not a native command-line shell for Windows (cmd is), there are a few additional wrinkles.
- ▶ Certain programs are designed to run within a console window. Those need to be prefixed with *winpty*. So if you want Python interactive shell:
 - ◆ `winpty python`
- ▶ Pay attention to your directory path.
 - ◆ In git-bash, full path starts with `/c/`.
 - ◆ In cmd (Windows native), it is `C:\...`
 - ◆ In Python, full path can be written as `'C:/...'` or `'C:\\...'` or `r'C:\...'`.
- ▶ Not included:
 - ◆ `more` (use `less` instead)
 - ◆ `man` (you're going to have to Google)

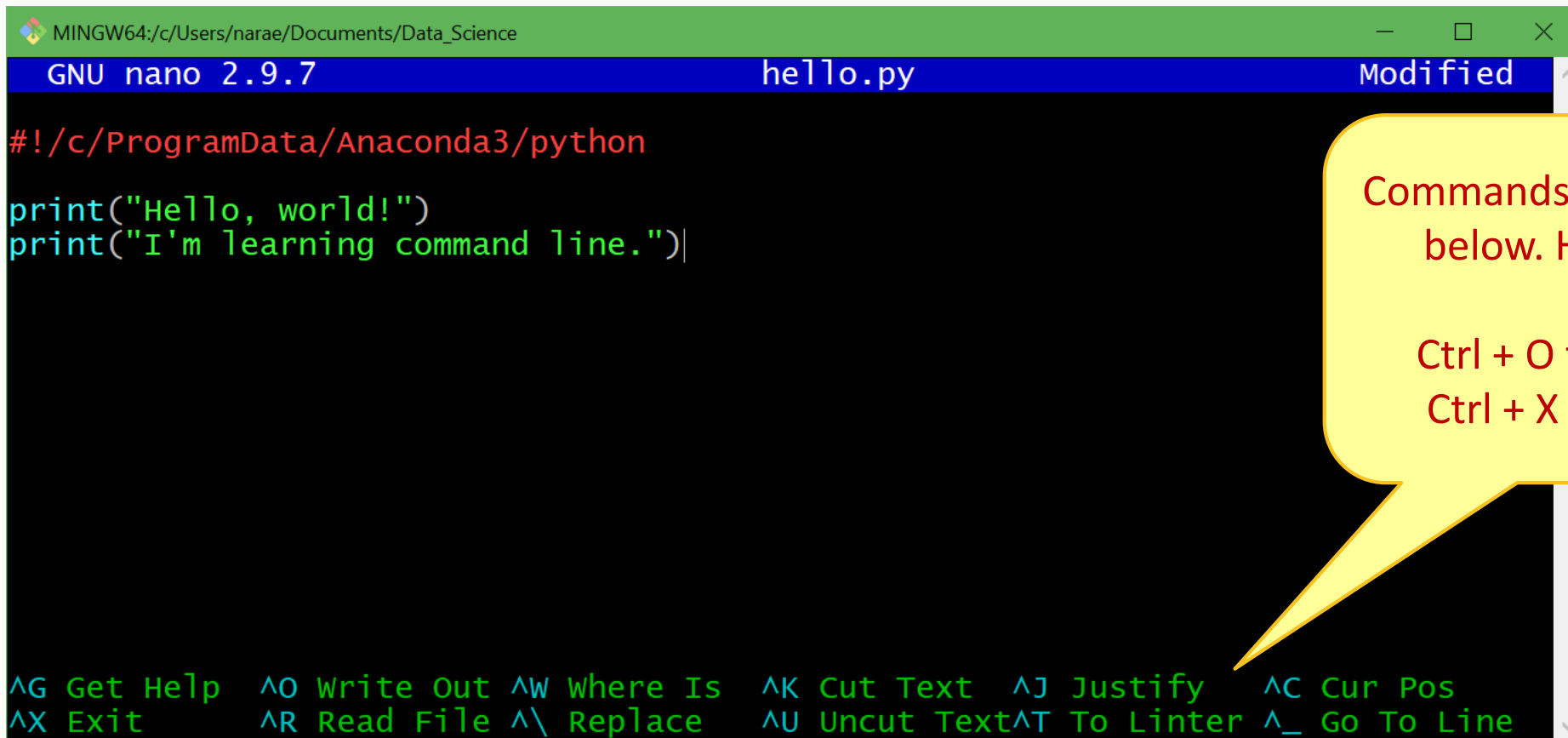
Mac users



- ▶ Add some aliases.
- ▶ Like in Windows, you should be able to launch any app that is found in your PATH.
- ▶ Surprise! You also have a handy command for launching *any* GUI application from command-line.
 - ◆ `open -a Application-Name`
 - ◆ <http://osxdaily.com/2007/02/01/how-to-launch-gui-applications-from-the-terminal/>

nano

- ▶ **nano** is a simple command-line based editor. It is found on all Linux distros.
 - ◆ Already present on Macs, and also part of Windows git Bash.



```
MINGW64:/c/Users/narae/Documents/Data_Science
GNU nano 2.9.7 hello.py Modified
#!/c/ProgramData/Anaconda3/python
print("Hello, world!")
print("I'm learning command line.")

^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify  ^C Cur Pos
^X Exit      ^R Read File ^\ Replace  ^U Uncut Text ^T To Linter ^_ Go To Line
```

Commands are listed below. Handy!

Ctrl + O to save
Ctrl + X to exit

Running python script from command-line

1. `python hello.py`

- ◆ Assuming python is in your \$PATH, and hello.py is in your current working directory

2. `hello.py`

- ◆ Assuming your current working directory is in your \$PATH. If not, you should execute `./hello.py`

- ◆ Assuming your script begins with a line (called 'shebang' line):

`#!/systempath/to/python`

- ◆ In my case, it's `#!/c/ProgramData/Anaconda3/python`
- ◆ If your path contains a SPACE... tough luck! (Just kidding, there are ways to handle this.)

Piping and I/O redirection

- ▶ **Piping and I/O redirection** make command-line ever so powerful.
- ▶ For people working mainly with text data (us!), piping enables us to manipulate data on the fly.
 - ◆ `hello.py > out.txt` redirect output to file
 - ◆ `hello.py | wc` pipe output to another application
 - ◆ `hello.py | wc > out.txt` daisy chain!

Also:

- ◆ `<` read in from a file input
- ◆ `>>` *append* to existing file rather than overwriting

Download two files

- ▶ Alice's Adventures in Wonderland

- ◆ <http://www.gutenberg.org/ebooks/11>
- ◆ Download the Plain Text UTF-8 version.
- ◆ Rename the file to "alice.txt"

- ▶ ENABLE word list from Peter Norvig's site:

- ◆ <http://norvig.com/ngrams/>
- ◆ Download "enable1.txt".

← Save them onto your Desktop.

← Then, within bash shell, move the files into your Data_Science directory. (Wait if you are not sure how this is done.)

Files in your Data_Science directory

```
MINGW64:/c/Users/narae/Documents/Data_Science
narae@T450s MINGW64 ~/Documents
$ cd Data_Science/

narae@T450s MINGW64 ~/Documents/Data_Science
$ ls
Class-Practice-Repo/  HW2-Repo/  planets/
Corpus-Resources/    Inaugural-Address-Project/  real_linguistics_data/
HW1-Repo/             foo/

narae@T450s MINGW64 ~/Documents/Data_Science
$ mv ~/Desktop/alice.txt .

narae@T450s MINGW64 ~/Documents/Data_Science
$ mv ~/Desktop/enable1.txt .

narae@T450s MINGW64 ~/Documents/Data_Science
$ ls
Class-Practice-Repo/  Inaugural-Address-Project/  planets/
Corpus-Resources/    alice.txt                    real_linguistics_data/
HW1-Repo/            enable1.txt
HW2-Repo/            foo/

narae@T450s MINGW64 ~/Documents/Data_Science
$ |
```

Examining a text file

▶ `ls (-lahF)`

- ◆ Displays file info

▶ `wc`

- ◆ Displays line count, word count, and character count

▶ `head -n`

- ◆ Displays initial n lines

▶ `tail -n`

- ◆ Displays last n lines

```
MINGW64:~/Documents/Data_Science
narae@X1Yoga MINGW64 ~/Documents/Data_Science
$ ls -l enable1.txt
-rw-r--r-- 1 narae 197121 1916146 Mar 19 12:39 enable1.txt

narae@X1Yoga MINGW64 ~/Documents/Data_Science
$ ls -lh enable1.txt
-rw-r--r-- 1 narae 197121 1.9M Mar 19 12:39 enable1.txt

narae@X1Yoga MINGW64 ~/Documents/Data_Science
$ wc enable1.txt
172819 172820 1916146 enable1.txt

narae@X1Yoga MINGW64 ~/Documents/Data_Science
$ wc alice.txt
3736 29465 173595 alice.txt

narae@X1Yoga MINGW64 ~/Documents/Data_Science
$ head enable1.txt
aa
aah
aahed
aahing
aahs
aal
aalii
aaliis
aals
aardvark

narae@X1Yoga MINGW64 ~/Documents/Data_Science
$ tail -5 enable1.txt
zymotic
zymurgies
zymurgy
zyzzyva
zyzzyvas

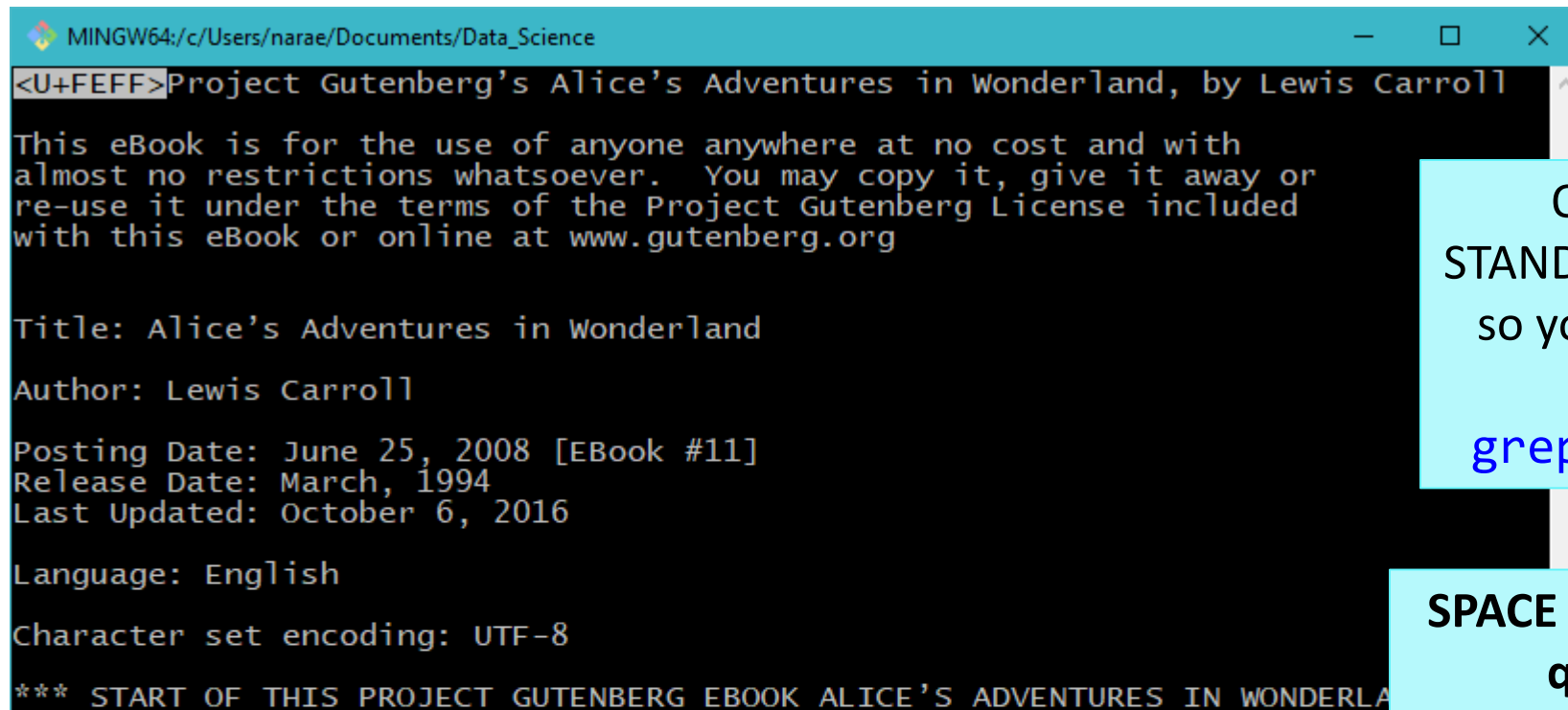
narae@X1Yoga MINGW64 ~/Documents/Data_Science
$ head -5 alice.txt
Project Gutenberg's Alice's Adventures in Wonderland, by Lewis Carroll

This eBook is for the use of anyone anywhere at no cost and with
almost no restrictions whatsoever. You may copy it, give it away or
re-use it under the terms of the Project Gutenberg License included

narae@X1Yoga MINGW64 ~/Documents/Data_Science
$
```


more or less

- ▶ **more** (and **less**) through a text file content, one screen-full at a time. Press **SPACE** for next page, **q** to quit.
 - ◆ Windows users: only **less** is available on git bash.



```
MINGW64:/c/Users/narae/Documents/Data_Science
<U+FEFF>Project Gutenberg's Alice's Adventures in Wonderland, by Lewis Carroll
This eBook is for the use of anyone anywhere at no cost and with
almost no restrictions whatsoever. You may copy it, give it away or
re-use it under the terms of the Project Gutenberg License included
with this eBook or online at www.gutenberg.org

Title: Alice's Adventures in Wonderland
Author: Lewis Carroll
Posting Date: June 25, 2008 [EBook #11]
Release Date: March, 1994
Last Updated: October 6, 2016

Language: English
Character set encoding: UTF-8

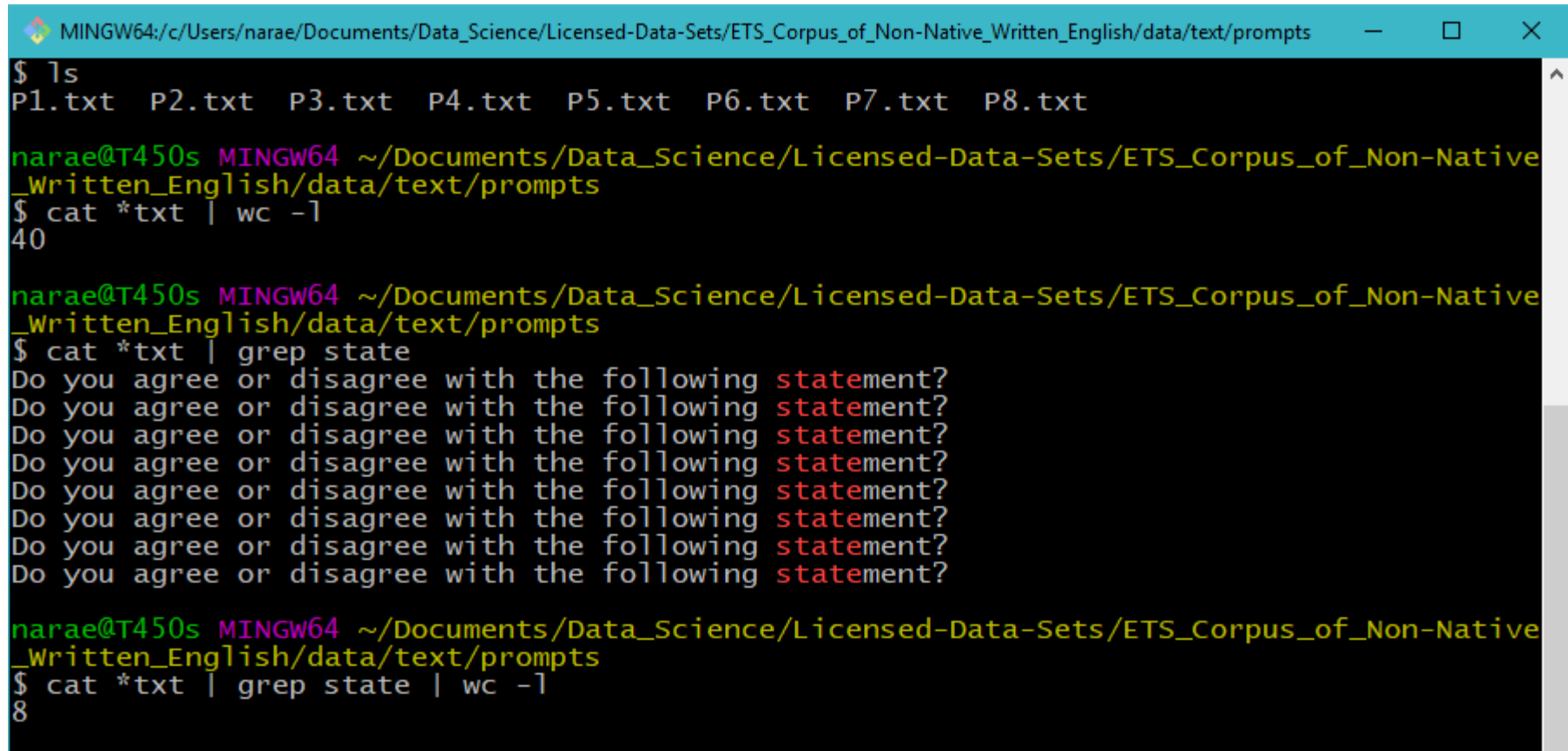
*** START OF THIS PROJECT GUTENBERG EBOOK ALICE'S ADVENTURES IN WONDERLA
```

Often, you **pipe** your STANDARD OUTPUT into more, so you can look through the result, e.g.,
`grep 'q' words | less`

SPACE for next page
q to quit

cat

- ▶ **cat** concatenates text file content and prints on the standard output.
 - ◆ Often used as the first step of piping.
 - ◆ Also useful in concatenating multiple file contents.



```
MINGW64:/c/Users/narae/Documents/Data_Science/Licensed-Data-Sets/ETS_Corpus_of_Non-Native_Written_English/data/text/prompts
$ ls
P1.txt P2.txt P3.txt P4.txt P5.txt P6.txt P7.txt P8.txt

narae@T450s MINGW64 ~/Documents/Data_Science/Licensed-Data-Sets/ETS_Corpus_of_Non-Native_Written_English/data/text/prompts
$ cat *txt | wc -l
40

narae@T450s MINGW64 ~/Documents/Data_Science/Licensed-Data-Sets/ETS_Corpus_of_Non-Native_Written_English/data/text/prompts
$ cat *txt | grep state
Do you agree or disagree with the following statement?
Do you agree or disagree with the following statement?
Do you agree or disagree with the following statement?
Do you agree or disagree with the following statement?
Do you agree or disagree with the following statement?
Do you agree or disagree with the following statement?
Do you agree or disagree with the following statement?
Do you agree or disagree with the following statement?

narae@T450s MINGW64 ~/Documents/Data_Science/Licensed-Data-Sets/ETS_Corpus_of_Non-Native_Written_English/data/text/prompts
$ cat *txt | grep state | wc -l
8
```

grep!!!

▶ grep

- ◆ Searches each line in text for **regular expression** match
- ◆ Excellent intro: <http://www.softpanorama.org/Tools/grep.shtml>

▶ grep -E

- ◆ "extended" regular expression search
- ◆ same as **egrep**

▶ grep -P

- ◆ Only on git-Bash & Linux: **Mac users should use pcregrep instead**
- ◆ Accepts **perl-style** regular expressions
- ◆ Perl-style = Python-style! Can use `\s`, `\d` etc.

```
MINGW64:/c/Users/narae/Documents/Data_Science
narae@T450s MINGW64 ~/Documents/Data_Science
$ grep '^o.*o$' enable1.txt
obligato
obligato
ocotillo
octavo
oho
oleo
olio
oloroso
onto
oratorio
ordo
oregano
ortho
orzo
ostinato
otto
outdo
utecho
outgo
ouzo
overdo
ovoio
oxo

narae@T450s MINGW64 ~/Documents/Data_Science
$ grep '^a.*z$' enable1.txt
abuzz
adz

narae@T450s MINGW64 ~/Documents/Data_Science
$ grep -P '[aeiou]{5,}' enable1.txt
cooeeing
miaoued
miaouing
queueing

narae@T450s MINGW64 ~/Documents/Data_Science
$ |
```

Non-Windows folks
will notice something's
wrong. What?



Words with 5+
consecutive "vowel"s

CRLF vs. LF



- ▶ Windows uses `'\r\n'` ("CRLF") as line ending, Mac/Linux `'\n'`
- ▶ Our 'enable1.txt' file has Windows-style, CRLF line ending:

```
narae@X1Yoga MINGW64 ~/Documents/Data_Science
$ file enable1.txt
enable1.txt: ASCII text, with CRLF line terminators
```

file displays encoding, line ending, etc. (but not infallible)

- ▶ How to convert:
 - ◆ <https://www.cyberciti.biz/faq/howto-unix-linux-convert-dos-newlines-cr-lf-unix-text-format/>
 - ◆ Use `dos2unix`
 - ◆ Use `tr`
 - ◆ Use one-liner `perl`
 - ◆ etc.

grep is better in color

- ▶ You might want to colorize your grep output.
- ▶ I have `grep` aliased to use color & perl-style regex in my `.bash_profile` configuration file:

```
MINGW64:/c/Users/narae/Documents/Data_Science

narae@X1Yoga MINGW64 ~/Documents/Data_Science
$ grep '[aeiou]{5,}' enable1.txt
cooeeing
miaoued
miaouing
queueing

narae@X1Yoga MINGW64 ~/Documents/Data_Science
$ cat ~/.bash_profile
alias more='less'
alias grep='grep -P --color'
```

grep -i, -v

- ▶ **grep -i**
 - ◆ ignores case
- ▶ **grep -v**
 - ◆ prints lines that DO NOT match

```
narae@T450s MINGW64 ~/Documents/Data_Science
$ grep -i 'q' enable1.txt | grep -v 'u'
faqir
faqirs
qaid
qaidS
qanat
qanats
qat
qats
qindar
qindarka
qindars
qintar
qintars
qoph
qophs
qwerty
qwertys
sheqalim
sheqel
tranq
tranqs

narae@T450s MINGW64 ~/Docu
$ |
```

```
MINGW64:/c/Users/narae/Documents/Data_Science

narae@T450s MINGW64 ~/Documents/Data_Science
$ cat enable1.txt | grep -Pv '[aeiouy]'
brr
brrr
crwth
crwthS
cwm
cwms
hm
hmm
mm
nth
pfft
phpht
pht
psst
sh
shh
tsk
tsks
tsktsk
tsktsks
```

grep and piping, together

MINGW64:/c/Users/narae/Documents/Data_Science

```
unwarrantable  
unwatchable  
unwearable  
unwinnable  
unworkable
```

```
narae@T450s MINGW64 ~/Documents/Data_Science  
$ grep '^un.*able$' enable1.txt | wc -l  
213
```

Pipe into wc -l to count

```
narae@T450s MINGW64 ~/Documents/Data_Science  
$ grep '^un.*able$' enable1.txt > able.txt
```

Write out to a file

```
narae@T450s MINGW64 ~/Documents/Data_Science  
$ tail -5 able.txt  
unwarrantable  
unwatchable  
unwearable  
unwinnable  
unworkable
```

Take a look at the last 5 lines of file

```
narae@T450s MINGW64 ~/Documents/Data_Science  
$ grep '^in.*able$' enable1.txt >> able.txt
```

Append new search result to file

```
narae@T450s MINGW64 ~/Documents/Data_Science  
$ tail -5 able.txt  
invariable  
investable  
inviabile  
inviolable  
invulnerable
```

Take a look at the last 5 lines of file

```
narae@T450s MINGW64 ~/Documents/Data_Science  
$ wc -l able.txt  
316 able.txt
```

File is now longer

```
narae@T450s MINGW64 ~/Documents/Data_Science  
$ |
```

grep -C n

▶ grep -C 2

- ◆ prints context: 2 lines before and after

← capital C!

```
narae@X1Yoga MINGW64 ~/Documents/Data_Science
$ grep -iC 2 "curious" alice.txt
* * * * *
* * * * *
```

```
'What a curious feeling!' said Alice; 'I must be shutting up like a telescope.'
```

```
--
her eyes; and once she remembered trying to box her own ears for having
cheated herself in a game of croquet she was playing against herself,
for this curious child was very fond of pretending to be two people.
'But it's no use now,' thought poor Alice, 'to pretend to be two people!
why, there's hardly enough of me left to make ONE respectable person!'
```

```
--
CHAPTER II. The Pool of Tears
```

```
'Curiouser and curiouser!' cried Alice (she was so much surprised, that
for the moment she quite forgot how to speak good English); 'now I'm
opening out like the largest telescope that ever was! Good-bye, feet!'
```

```
--
It was high time to go, for the pool was getting quite crowded with the
birds and animals that had fallen into it: there were a Duck and a Dodo,
a Lory and an Eaglet, and several other curious creatures. Alice led the
way, and the whole party swam to the shore.
```

```
--
always growing larger and smaller, and being ordered about by mice and
rabbits. I almost wish I hadn't gone down that rabbit-hole--and yet--and
yet--it's rather curious, you know, this sort of life! I do wonder what
CAN have happened to me! When I used to read fairy-tales, I fancied that
kind of thing never happened, and now here I am in the middle of one!
```

```
--
by another footman in livery, with a round face, and large eyes like a
frog; and both footmen, Alice noticed, had powdered hair that curled all
over their heads. She felt very curious to know what it was all about,
and crept a little way out of the wood to listen.
```


Not done with grep

- ▶ ... grep continues next class.

(Fun fact: I had prepared 47 slides for today's class.)

Wrapping up

▶ To-do #12

- ◆ Fun with big(ish) data -- the Yelp Dataset! <https://www.yelp.com/dataset/challenge>
- ◆ Downloading data alone takes about 25 minutes. Allocate enough time for this assignment, especially if you are new to command line.

▶ Next class

- ◆ More command line, grep, bash shell scripting
- ◆ Supercomputing at CRC
- ◆ More on machine learning