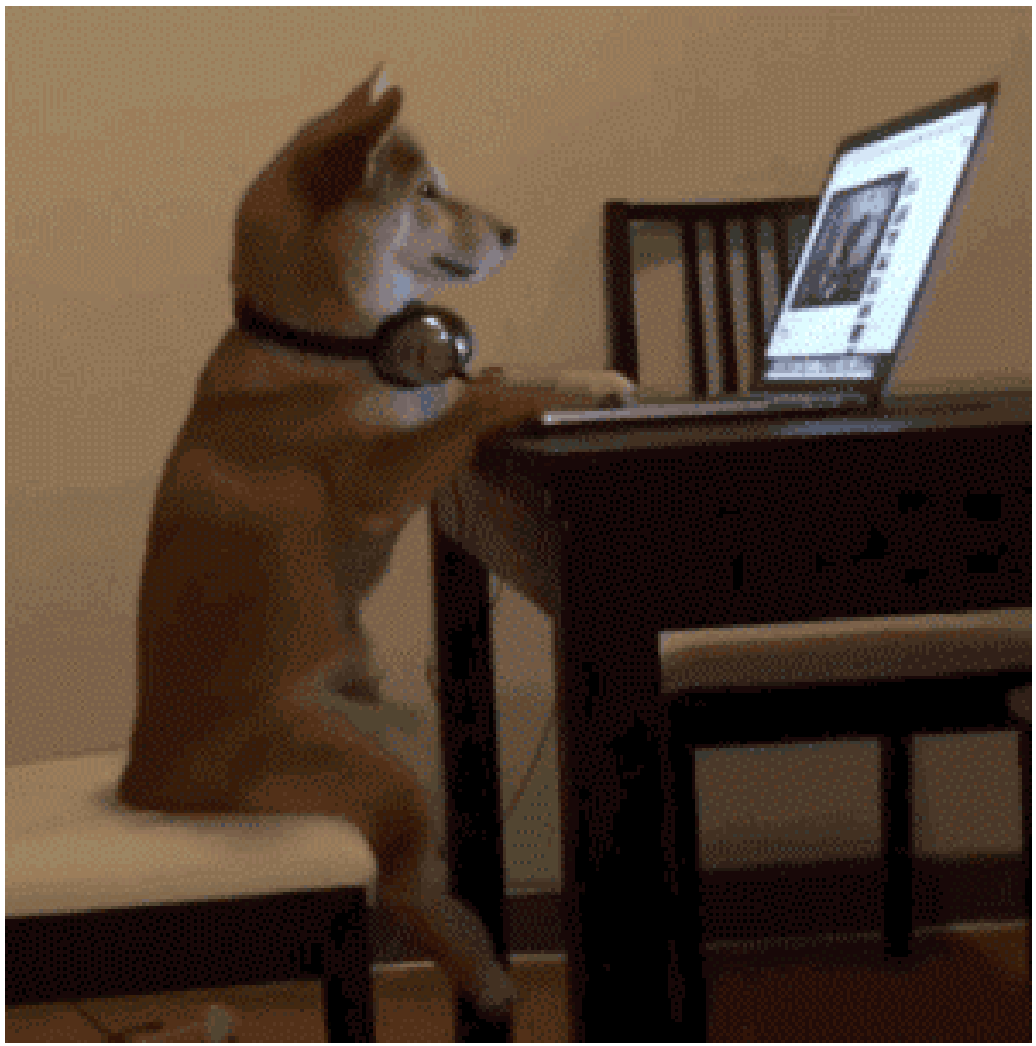
The Power of the Command Line

Jae Yeon Kim

Objectives



- Do you really know how POWERFUL your laptop is?

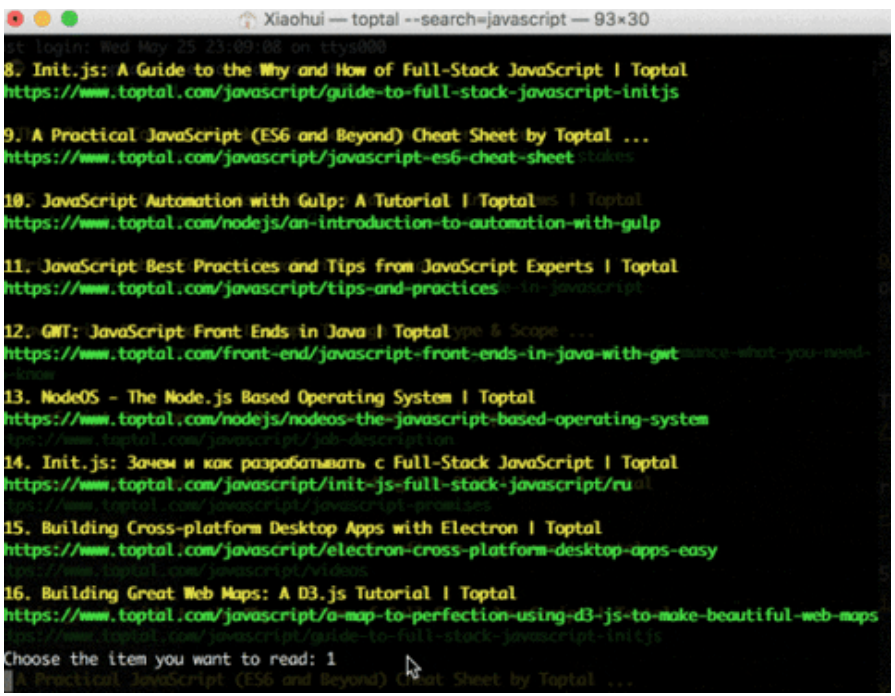
- It can do what you want want to do if you know how to teach it to do it

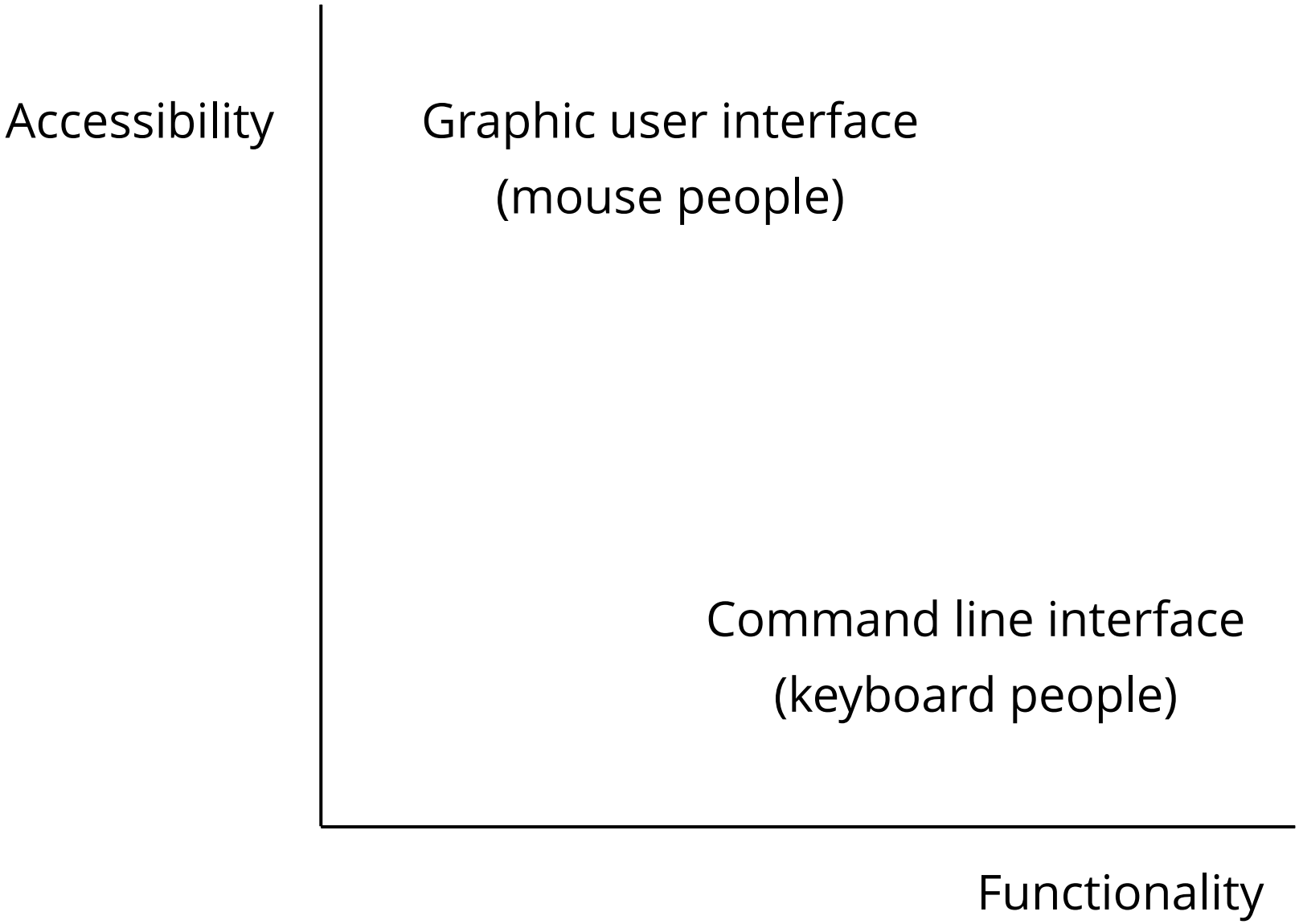
- That's basically programming

How to talk to your computer?

Graphic user interface

Command line interface




Accessibility
Graphic user interface
(mouse people)
Command line interface
(keyboard people)
Functionality

“Graphical user interfaces make easy tasks easy, while command line interfaces make difficult tasks possible."

- William Shotts the author of The Linux Command Line

Motivating example

Problem:

You have 100 documents (=text files). One of them contains the word "COVID" and you want to fine out which one is. How can you do that?

Using GUI, you have two options.

1. Do it by yourself

2. Hire undergraduate assistants

Good luck!

Or you can do it using the command line.

```
1 grep -rnw -e "COVID"
```

Note two things here.

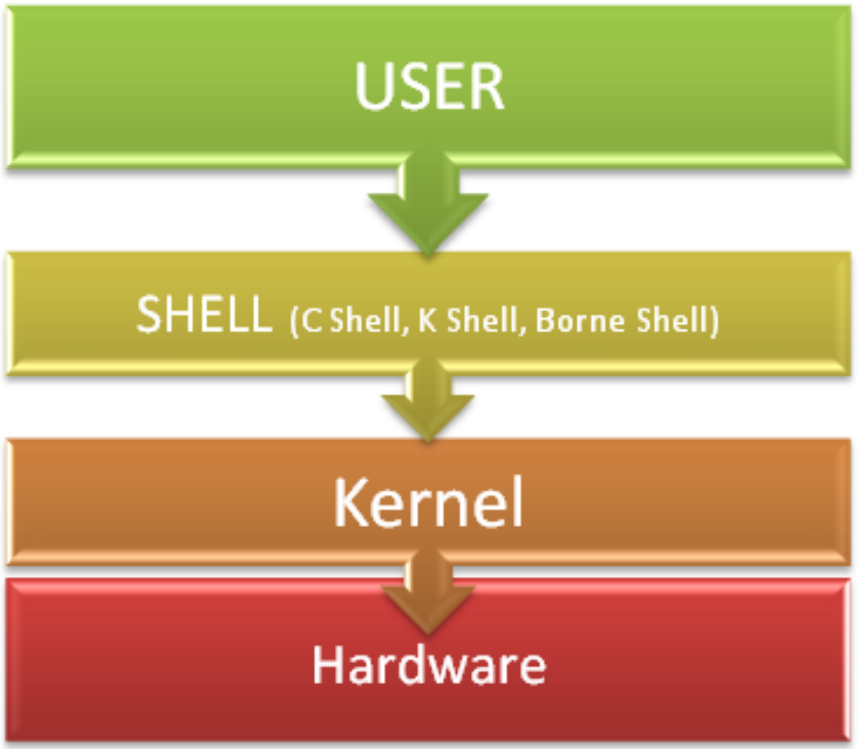
1. Simplicity: It's just one-liner solution.

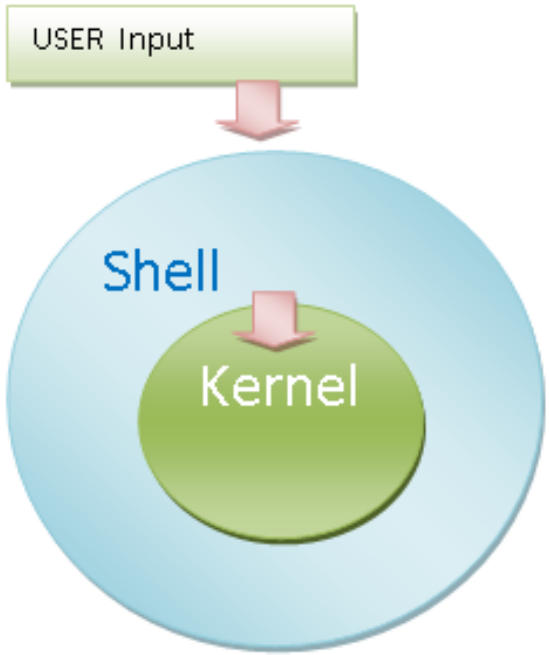
2. Efficiency: The solution works even if the scale goes up (1 million documents).

One more: you can make this process entirely reproducible!

```
# Create test file that contains the word "test"
echo "test" > test

# Make 100 duplicates
for i in {1..100}; do cp test "test_$i"; done

# Append "COVID" to test_100
echo "COVID" >> test_100

# Check
cat test_100

# Find the document that contains the word "COVID"
grep -rnw -e "COVID"

# Remove all of these files
rm test_* test
```

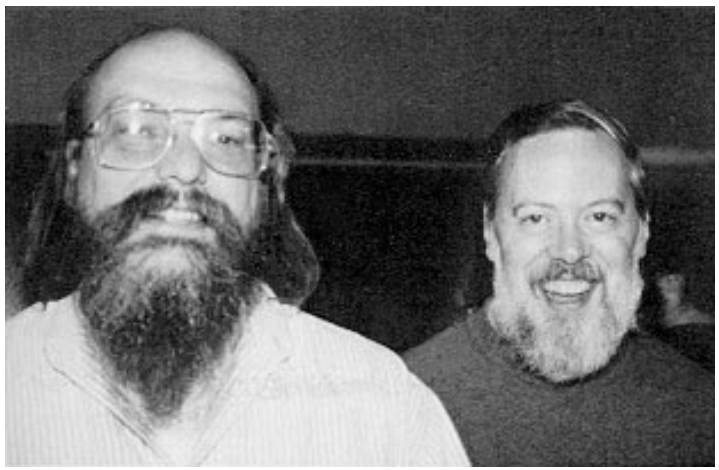
Shell Programming





1. Technically, this is what it means by using the command-line. You're using shell to control a comptuer via kernel (the hub of the operating system).

2. Bash is one of the most popular *Unix* shells.



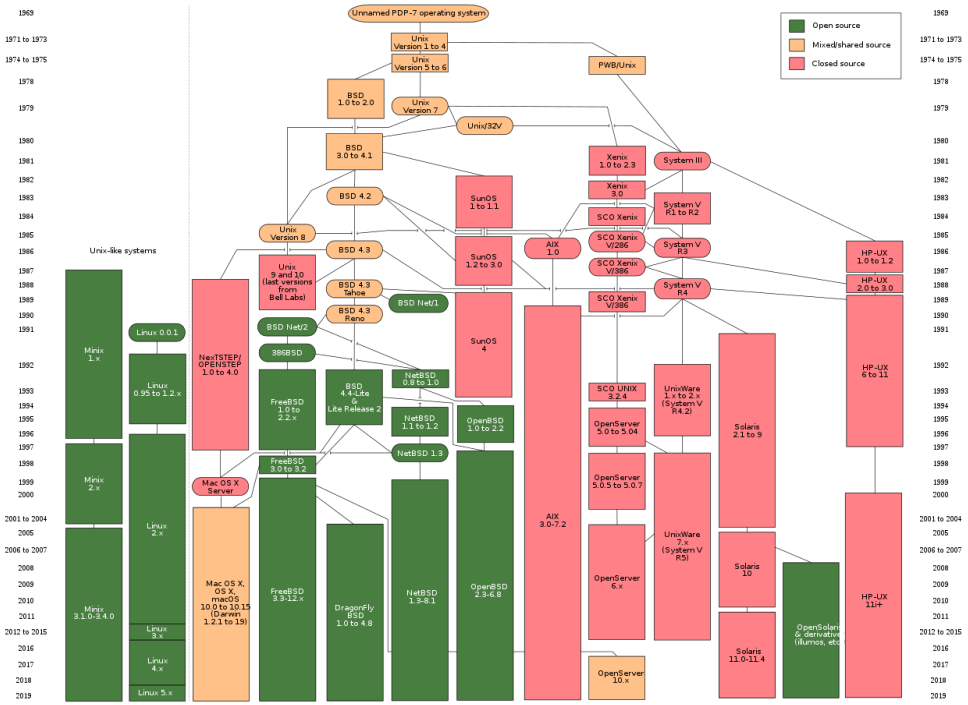
Ken Thompson and Dennis Ritchie, key proponents of the Unix philosophy

- Heard of UNIX?

It's an OS + a set of utilities.

- Developed at Bell Labs (1969-1971). Old, but not outdaded.

- Lots of **UNIX tools (command-line)** are still popular. Its philsophy ("Do One Thing And Do It Well")---minimalist, modular software development---is highly influential.



- Linux and Mac are UNIX descedants. No installation issues.

- Not Windows. Windows users need to install either GitBash (basic use) or Windows Subsystem/PowerShell (intermediate use).

- If you use Windows v. < 10 then I recommen installing virtual box. (For more information, see ref book chatpter 3)

Let's learn commands

Open your terminal

```
1 <user name>$<machine name>
2 jae@jae-X705UDR:~$
```

- jae: a specific user name
- jae-X705UDR: your computer/server name
- ~: current directory (~ = home)
- $: a **prompt**, which shows us that the shell is waiting for input; your shell may show something more elaborate.

Find your user name: whoami

```
1 jae@jae-X705UDR:~$ whoami
2
3 # Should be your user name
4 jae
```

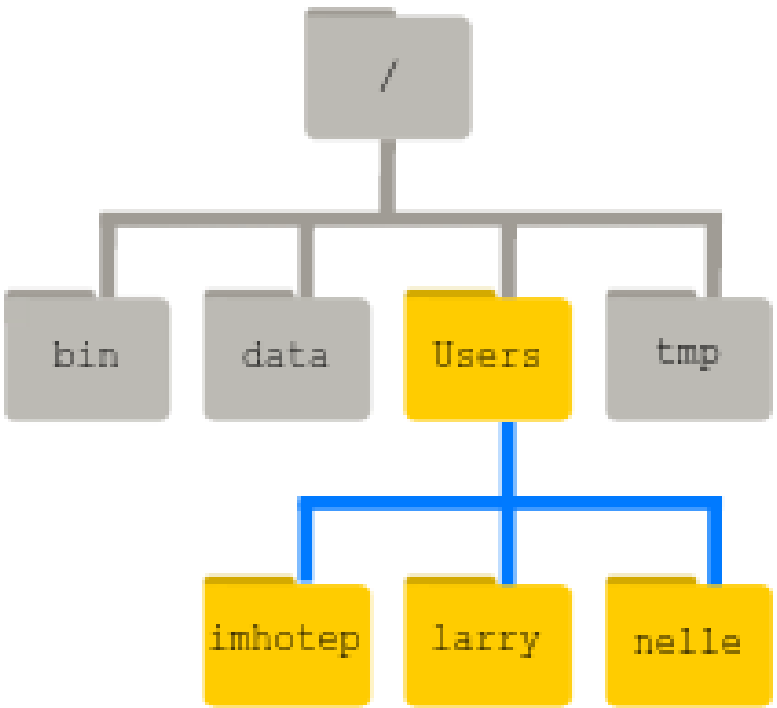
More specifically, when we type whoami the shell:

1. finds a program called whoami,
2. runs that program,
3. displays that program's output, then
4. displays a new prompt to tell us that it's ready for more commands.

Accessing to the Internet: Why bother webbrowser

```
1 $ ping 8.8.8.8 # acccess to Google
  Domain Name Server
2 $ ping -t 8.8.8.8 # acccess to
  Google Domain Name Server
3 $ sudo apt-get install wget #
  Install wget utility
4 $ wget
  https://download1.rstudio.org/deskto
  p/bionic/amd64/rstudio-1.4.1103-
```

Navinging the file system: an upside-down tree (hierarhcy)



Listing files: ls

```
1 $ pwd # Current working directory; Show where you
  are in the file system
2 $ ls # list files
3 # ls -l # list files in a long format
4 # ls -F / # list files by type (F) and the type
  we use here is / (directory)
```

1. Remember logic not documentation (you can always get this by <command> --help>
2. Famialize yourself this this syntax <command> -<flag>. If command is Verb, then flag is Adverb.

Finding files: cd + find

```
1 $ cd Downloads/
2 $ find *.pdf
```

1. `cd`: change directory (dont' type everything; use autocompletion by TAB)
2. `find`: find files
3. * (any number of characters)

k.pdf

ki.pdf

kim.pdf

Finding files: cd + find

```
1 $ find *.pdf | less # Show output
  incrementally
2 $ ls -t | find *.pdf | head -n1 # Show only
  the most recent downloaded PDF document
```

1. `|`: chain commands
2. `less`: print output incrementally
3. `head`: show the first ten lines of the output

Creating files

```
1 $ mkdir exercise # Create an empty directory
  named exercise
2 $ cd exercuse | touch test ; ls # Let's move to
  exercise subdirectory and create a file named
  test
3 $ cat test # Read test
4 $ echo "something" test ; cat test # Let's add
  something there. > = overwrite
5 $ echo "anything" >> test ; cat test # Let's
  appned anything >> = append
6 $ nano test # if you want to use editor
```

Copying and renaming files

1. Make one duplicate

```
1 $ cp test test_1 # copy test to test_1
```

2. Make 100 duplicates (using for loop)

```
1 $ for i in {1..100}; do cp test
  "test_$i"; done
```

3. Remove all duplicates

```
1 $ rm test_*
```

Moving files: mv

```
1  # Create two directories
2  $mkdir exercise_1 ; mkdir exercise_2
3
4  # Check whether they were indeed created
5  $find exer*
6
7  # Create an empty file
8  $touch exercise_1/test
9
10 # Move to exercise_1 and check
11 $cd exercise_1 ; ls
12
13 # Move this file to exercise_2
14 $mv test ../exercise_2
15
16 # Move to exercise_2 and check
17 $cd exercise_2 ; ls
```

Key syntax: [mv] [source] [destination]

Working with csv files

```
1 # Download a CSV file (Forbes World's Billionaires lists from 1996-
  2014).
2 $wget https://corgis-
  edu.github.io/corgis/datasets/csv/billionaires/billionaires.csv
3
4 # Read the first two lines
5 $cat billionaires.csv | head -n2
```

1. Figure out the size of the csv file (# of rows)

```
1 wc -l billionaires.csv
```

- wc prints newline, word, and byte counts for each file. If you run wc without -l flag, you get the following: 2615 (line) 20433 (word) 607861 (byte) billionaires.csv

- Figuring out the number of columns is a bit more complex.

Working with csv files

2. Filtering rows

```
1 wc -l billionaires.csv
```

- Filter rows that contain word "China" and count the number of these rows

User roles and file permissions

1. If you need admin access, use sudo. For instance, sudo apt-get install <package name>.

2. To run a Shell script (.sh), you need to change its file mode. You can make the script executable by chmod +x <Shell script>

Shell script examples

```
1 #!/bin/sh # Stating this is a Shell script.
2
3 mkdir /home/jae/Downloads/pdfs # Obviously, in
  your case this file path should be incorrect.
4
5 cd Download
6
7 cp *.pdf pdfs/
8
9 echo "Copied pdfs"
```

Here I show how to write a Shell script that creates a subdirectory called /pdfs under /Download directory, then find PDF files in /Download and copy those files to pdfs

Shell script examples

```
#!/bin/bash

R --version | head -n 1

git --version | head -n 1

python --version | head -n 1

nano --version | head -n 1

pdflatex --version | head -n 1

```

Can you tell me what it does?