Unix Fundamentals

Learning to communicate with your computer

// FLATIRON SCHOOL

Agenda

1. What is an operating system?
2. What is a file system?
3. What is Unix?
4. How do you interact with Unix?
5. Practice Problems

1. What is ***an operating system?***

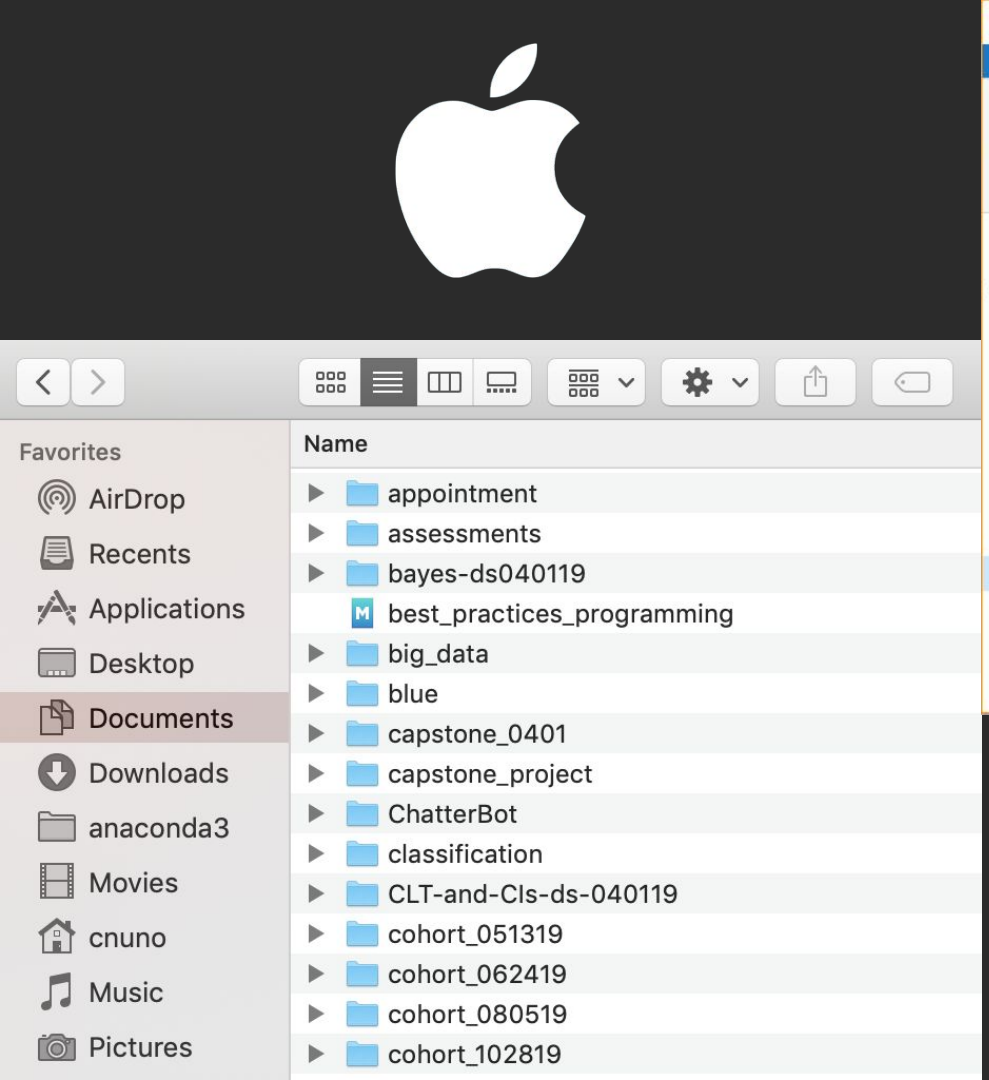

OS

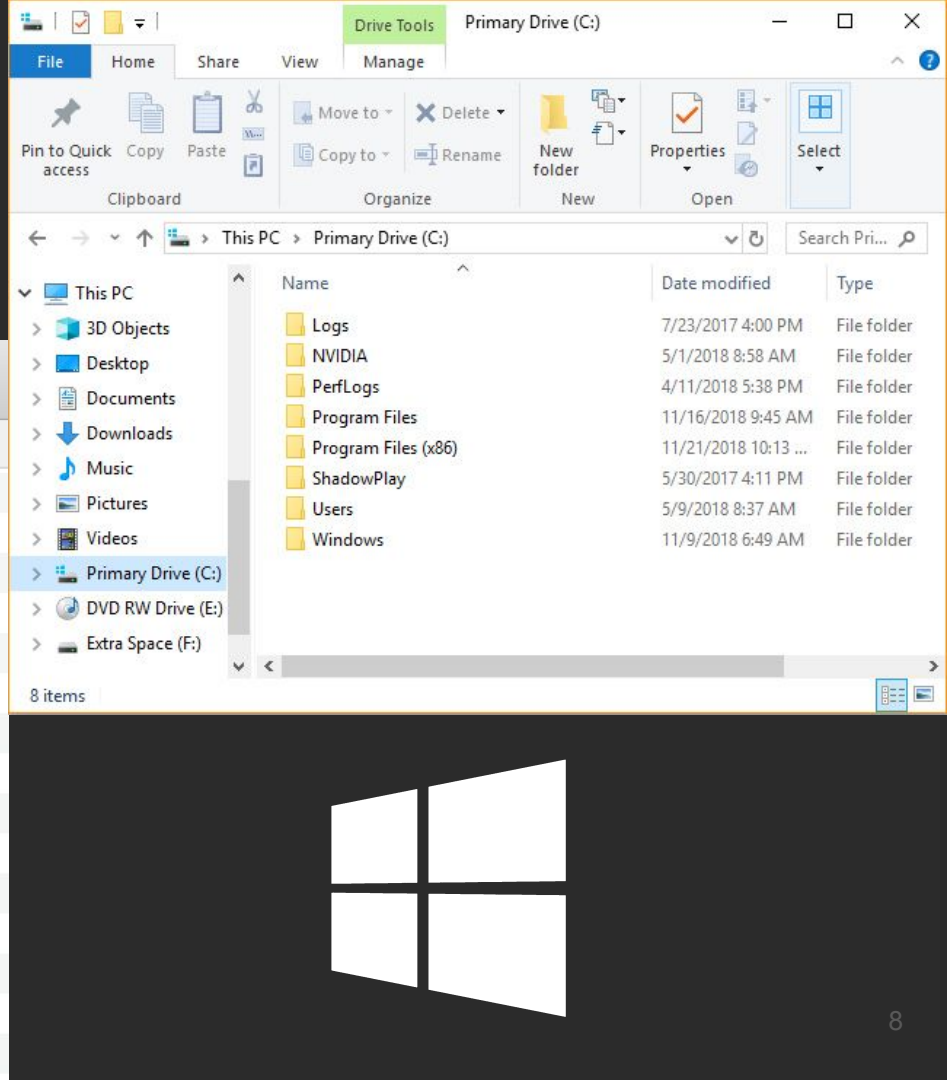
“An operating system (OS) is system software that manages computer hardware, software resources, and provides common services for computer programs.”

— *Wikipedia*

2. What is ***a file system?***


2008 Taxes
2011 Taxes
Store Credit Cards
EverBank
Mazda3
Home Purchase Docs
Home Insurance
Major Purchases
Comcast
Water
Stafford Loan
Grad Plus Loan
Health Insurance
Medical Receipts
In Case of Emergency
Taxes
Financial
Auto
Home
Utilities
Education
Medical
ICE


Favorites
AirDrop
Recents
Applications
Desktop
Documents
Downloads
anaconda3
Movies
cnuno
Music
Pictures
Name
appointment
assessments
bayes-ds040119
best_practices_programming
big_data
blue
capstone_0401
capstone_project
ChatterBot
classification
CLT-and-CIs-ds-040119
cohort_051319
cohort_062419
cohort_080519
cohort_102819


Drive Tools
Primary Drive (C:)
File
Home
Share
View
Manage
Pin to Quick access
Copy
Paste
Move to
Delete
Copy to
Rename
New folder
Properties
Select
Clipboard
Organize
New
Open
This PC
Primary Drive (C:)
Search Pri...
This PC
3D Objects
Desktop
Documents
Downloads
Music
Pictures
Videos
Primary Drive (C:)
DVD RW Drive (E:)
Extra Space (F:)
Name
Date modified
Type
Logs 7/23/2017 4:00 PM File folder
NVIDIA 5/1/2018 8:58 AM File folder
PerfLogs 4/11/2018 5:38 PM File folder
Program Files 11/16/2018 9:45 AM File folder
Program Files (x86) 11/21/2018 10:13 ... File folder
ShadowPlay 5/30/2017 4:11 PM File folder
Users 5/9/2018 8:37 AM File folder
Windows 11/9/2018 6:49 AM File folder
8 items

"In computing, a file system...controls how data is stored and retrieved.

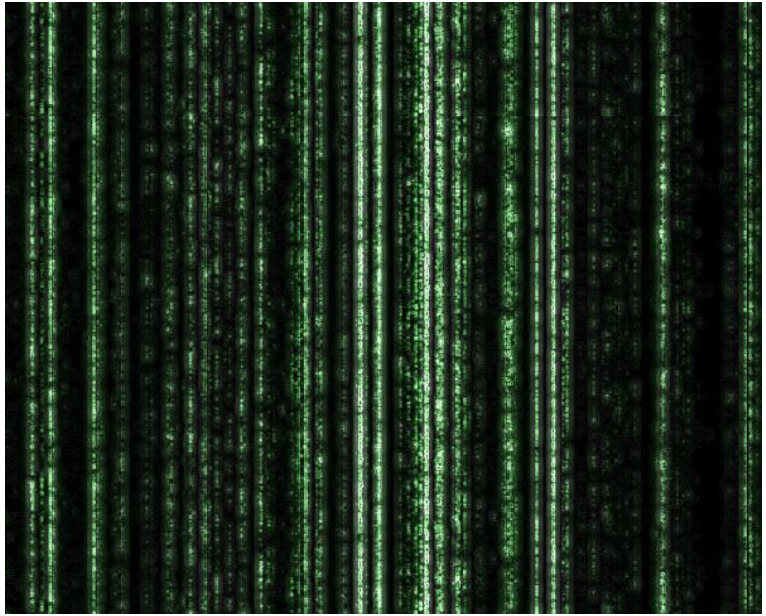
By separating the data into pieces and giving each piece a name, the information is easily isolated and identified.

Each [piece] of data is called a 'file'."

– *Wikipedia*

3. What is ***Unix?***

Fantasy



Reality

```
(learn-env) SF-US13603-MBA:cohort_102819 cnuno$ pwd
/Users/cnuno/Documents/cohort_102819
(learn-env) SF-US13603-MBA:cohort_102819 cnuno$ ls -l
total 0
drwxr-xr-x   7 cnuno  staff  224 Oct 22 16:56 daily_challenges_seattle-ds-102819
drwxr-xr-x  10 cnuno  staff  320 Oct 24 10:03 data_analysis_built_in_types
(learn-env) SF-US13603-MBA:cohort_102819 cnuno$ tree data_analysis_built_in_types/
data_analysis_built_in_types/
├── LICENSE
├── README.md
├── data_analysis.ipynb
├── raw_data
│   ├── EXTR_RPSale.csv
│   └── Real\ Property\ Sales.zip
└── visuals
    ├── README.md
    ├── clean_avg_sales_price_by_year.png
    ├── raw_avg_sales_price_by_year.png
    ├── raw_buyer_count_over_time.png
    ├── raw_pin_count_over_time.png
    ├── raw_sales_count_by_year.png
    └── raw_seller_count_over_time.png

2 directories, 12 files
(learn-env) SF-US13603-MBA:cohort_102819 cnuno$
```

"Unix is an operating system that provides a set of simple tools that each performs a limited, well-defined function, with a unified file system as the main means of communication, and a shell scripting and command language (the Unix shell) to combine the tools to perform complex workflows."

— *Wikipedia*

Unix Philosophy

"Write programs that do one thing and do it well.

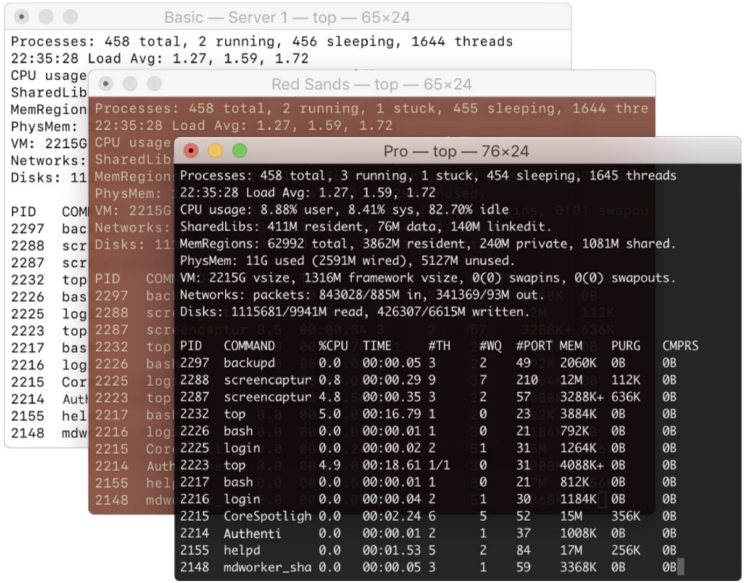
Write programs to work together.

Write programs to handle text streams, because that is a universal interface."

- ***Peter H. Salus***

4. How do you interact with ***Unix?***

From a Unix Shell that you access via the Terminal (also known as the Command Line Interpreter (CLI))



Bourne **A**gain **Sh**ell (Bash)

"Bash is a command processor that typically runs in a text window where the user types commands that cause actions. Bash can...read and execute commands from a file, called a shell script. Like all Unix shells, it supports filename globbing (wildcard matching), piping, here documents, command substitution, variables, and control structures for condition-testing and iteration." - Wikipedia

Z Shell (Zsh)

“The Z shell (Zsh) is a Unix shell that can be used as an interactive login shell and as a command interpreter for shell scripting...In June 2019, Apple announced that the forthcoming macOS Catalina (10.15) would adopt Zsh as the default shell, replacing Bash.” - Wikipedia

Streams as Standard Input and Output (I/O)

"Unix processes use I/O streams to read and write data. Processes read data from input streams and write data to output streams.

Streams are very flexible. For example, the source of an input stream can be a file, a device, a terminal, or even the output stream from another process." - Brian Ward

Primary I/O Streams	**Place where interaction occurs**
Standard Input	Keyboard
Standard Output	Terminal Window
Standard Error	Terminal Window

Example of Standard Input and Output

Commands are typed after the `$` (known as the shell prompt)

```
username$ <command goes here>
```

Commands are executed after hitting `ENTER/return`

Input → `username$ echo “Hello World”`

Output → `Hello World`

Common Commands (I/II)

- `pwd:` writes the absolute pathname of the current working directory to the standard output
- `ls:` displays the names (and additional info) of files contained within a directory
- `cd:` changes the current directory to the target directory
- `mkdir:` creates a new directory at specified target directory
- `export:` shell variables are marked for automatic export to the environment of subsequently executed commands

Common Commands (II/II)

- `cat:` reads files sequentially, writing them to the standard output
- `touch:` creates an empty file
- `rm:` removes both non-directory and directory files
- `echo:` writes arguments to the standard output
- `cp:` copies the contents of the source file to the destination target directory
- `mv:` either moves or renames the contents of the source file to the target directory

Shell Variables (I/II)

- The shell can store temporary variables, called shell variables, containing the values of text strings.

- Shell variables are very useful for keeping track of values in scripts, and some shell variables control the way the shell behaves.

- For example, the bash shell reads the `HOME` variable that enables you to return to your home directory at any time via `cd ~/`

- Source: ***Brian Ward***

Shell Variables (II/II)

- To assign a value to a variable, use the = sign:
  Input → `username$ FIRST_JOB=cashier`
  - This sets the value of the variable `FIRST_JOB` to `cashier`.

- To access `FIRST_JOB`, use `$FIRST_JOB`. To display its value to the screen, call the variable after `echo`:
  Input → `username$ echo $FIRST_JOB`
  Output → `cashier`

Aliases

- A shell feature that substitutes one string for another before executing a command
- A symbolic link is a file that points to another file or directory
- Potential to be used as an efficient shortcuts that save typing
- Potential to confuse yourself or coworkers when not properly documented

```
# color code directories & files
alias ls='ls -G'
export CLICOLOR=1
export
```

Bash Profile

- When you login to the Terminal, `.bash_profile` is executed to configure your shell before the initial command prompt.

- Enables you to set up aliases and define shell variables each time you launch the Terminal

- Source: [Stack Exchange]

.bash_profile

Users > cristiannuno > .bash_profile

```
# === Added by https://github.com/cenuno/sql_practice/ ===

# setup the class path for the JDBC Driver (i.e. for spark to connect to psql)
export PSQL_JAR=/usr/local/Cellar/apache-spark/2.4.4/libexec/jars/postgresql-4

# use the class path for the JDBC Driver each time we use pyspark
export PYSPARK_SUBMIT_ARGS='--conf spark.executor.extraClassPath=/usr/local/Ce

# make Java 1.8 your default Java Virtual Machine
export JAVA_HOME=/Library/Java/JavaVirtualMachines/adoptopenjdk-8.jdk/Contents

# locate spark home
export SPARK_HOME=/usr/local/Cellar/apache-spark/2.4.4/libexec

# add PySpark to PYTHONPATH
export PYTHONPATH=/usr/local/Cellar/apache-spark/2.4.4/libexec//python:/usr/lo

# === End of additions by https://github.com/cenuno/sql_practice/ ===
```

Homebrew

- The missing system package manager for Mac OS (or Linux)
- Homebrew installs the stuff you need that Apple (or your Linux system) didn't
- Homebrew installs packages to their own directory and then symlinks their files into /usr/local
- Allows you to download third party programs (i.e. python, r, git, postgresql, wget, etc.) to your machine
- Install Homebrew by following the directions listed here: https://brew.sh/

Vim

- Vim is a highly configurable text editor built to enable efficient text editing
- Great tool to know since it's the default text editor for git
- Often a source of frustration for beginners (see here)
- Launch a tutorial from the Terminal: `$ vimtutor` or play a free online game here at https://vim-adventures.com/
- Source: Vim Documentation

Bash Exercises

- Please see here for a great online tutorial on bash exercises: http://www.ee.surrey.ac.uk/Teaching/Unix/

Demonstration

Let’s find the terminal and try some things!

Exercises

1. Create a new directory called “flatiron_day1”.
2. Create a new file called “vim_notes.txt”.
3. Edit the file (in vim!) by writing into it: “In the command mode, use ‘h’ to move left, ‘j’ to move down, ‘k’ to move up, and ‘l’ to move right.”
4. Move the file into your flatiron_day1 folder.
5. Copy the folder to your Desktop. YOU WILL NEED TO USE `-rf`!
6. Delete the original folder.
7. List the contents of your Desktop to make sure that the flatiron_day1 folder is there!