LIN 127 - Text Processing and Corpus Linguistics

Week 1

Intro,
Unix, Shell,
Environment, Files

Who are we?

Instructor

Rob Voigt
robvoigt@ucdavis.edu

Assistant Professor
of Linguistics

Reader

Anthony Diaz
antdiaz@ucdavis.edu

PhD Student
in Linguistics

Who is this class for?

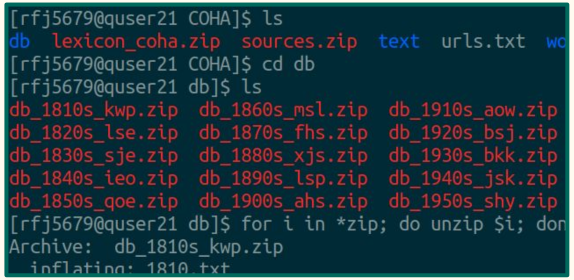
- Linguists, social scientists, humanists
- Little-to-no programming experience
- Applications to research

Goals

- Lots of hands-on practice
- Teach you how to teach yourself

Who is this class *not* for?

- Folks with lots of programming experience
- CS Majors (probably - email me if this is you)
- ECS 32A is similar in focus - what's different?
  - ECS 32A - broad, more CS-y
  - LIN 127 - narrow focus on applications to text, we will purposefully skip less-relevant stuff



```
[rfj5679@quser21 COHA]$ ls
db  lexicon_coha.zip  sources.zip  text  urls.txt  wo
[rfj5679@quser21 COHA]$ cd db
[rfj5679@quser21 db]$ ls
db_1810s_kwp.zip  db_1860s_msl.zip  db_1910s_aow.zip
db_1820s_lse.zip  db_1870s_fhs.zip  db_1920s_bsj.zip
db_1830s_sje.zip  db_1880s_xjs.zip  db_1930s_bkk.zip
db_1840s_ieo.zip  db_1890s_lsp.zip  db_1940s_jsk.zip
db_1850s_qoe.zip  db_1900s_ahs.zip  db_1950s_shy.zip
[rfj5679@quser21 db]$ for i in *zip; do unzip $i; don
Archive:  db_1810s_kwp.zip
  inflating: 1810.txt
```

What will we learn?

- Unix Command Line

  basic usage, navigating and editing, and tools for text

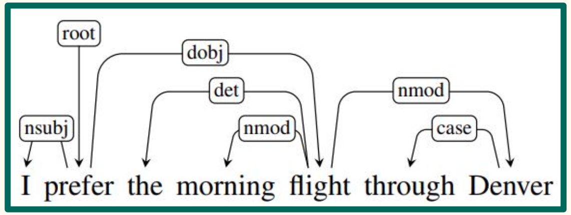
- Basic Python

  programming concepts, syntax, useful libraries for text

- Applications (as much as we have time)

  data munging, text analysis,

  web scraping, APIs



When and **where** will we see each other?

Here! Olson 146, MW 10:30-11:50am.

Rob's office hours: Wednesdays 1:30-2:30pm / by appt

Ed discussion board for questions - help each other out!

We will do a lot in person! Attendance is important -
you could learn a lot of this material by doing online tutorials,
the benefit here is personal guidance and asking questions.

Why are we doing this?

1. Get computationally “free” -
   GUIs only let you do things someone else decided on

2. Processing text data is useful for anyone’s research/work

3. This is the start of computational linguistics!
   large language models, conversational/generative AI,
   data science, web search, speech-to-text,
   “big data” language analysis, etc etc

How will we learn it?

Course structure and policies:

- Syllabus and Schedule
- Assignments
- Grading and Evaluation
- Agreements

Course Website

The main place to look for
syllabus, schedule, policies, and materials:

https://robvoigt.faculty.ucdavis.edu/courses/lin127

Discussion Forum

The main place to go for
questions, comments, concerns, discussion:

https://edstem.org/us/courses/90573/

(with apologies, but)
I *will not* respond to emails asking questions about the class.
You have to post on Ed. (you can post anonymously or privately)
If you have a question, someone else has it too.

Learning Structure - **In Class**

Some lecture / explanation

A fair amount of workshopping during class time

Periodic peer evaluations (focus on understanding and style)

Questions are **always** welcome - please stop me!

… just know with ~60 people I may sometimes have to move on, follow up on Ed and in OH

Learning Structure - **Assignments**

Generally 8-10 days from release to due date,

Largely programming, some qualitative aspects

Periodic in-class peer review / support / evaluation

Always basic core assignment, primary goal to pass tests; sometimes optional extra challenges if you're interested

Learning Structure - **Final Project**

Very open to possibilities!

Do something self-directed, fun, exciting!

Will discuss in detail again about halfway through - but talk with me at any point about ideas.

Grading and **Evaluation**

Emphasis on qualitative feedback where possible,
peer support, self-directed learning

We will record completion, passing base auto-graders;
then provide qualitative feedback on style/correctness

No comment = "good job!"

Grading and **Evaluation**

Letter grades at the end based on effortful completion, midterm and final self-evaluations (quick demo)

The point of this whole thing is for you to learn, period!

Flexibility

Class attendance and deadlines are ***strongly recommended***

but you're adults: missing either with a good reason (under your definition) is sufficient justification

if assignments are later than a week, we may not give feedback

Grading is (mostly) on your own scale relative to your own goals

This is a blessing and a curse! Prioritize appropriately.

What constitutes **strong performance**?

Effort and Engagement with learning.

Performance relative to *you*, not absolute performance.

Challenge yourself.

We may have a range of backgrounds and skill levels!

You are smart, you are adults -
I'll provide a structure, but it's ultimately on you.

What constitutes **strong performance**?

There is a lower bound:

Do basic reading/watching, complete core assignment
(make it work and pass auto-grader)

Be present, be engaged, be communicative
(if challenges arise you must inform us,
in comments in assignments or can private post on Ed)

What constitutes **strong performance**?

There is a lower bound:

> You cannot pass the class if you fail to submit any of Assignment 1, Midterm and Final Self-Evaluations
>
> You cannot pass the class if you do not complete at least 4 out 6 of Assignments 2-7

What constitutes **strong performance**?

There is no upper bound:

- Each week will have extra “relevant readings”
- Assignments sometimes have optional extra problems
- You can start working early on your final projects
- Plus whatever you can dream up

What constitutes **strong performance**?

There are no-nos:

- Doing the “bare minimum” (relative to you)
- Lack of communication
- Not doing the work
- Academic integrity / generative AI violations

Academic Integrity

I will attempt for this to be the only time I talk about this.

Here’s the line:

Talking with fellow students about questions you have so you understand better and work through problems
❤️🎆👍 **¡AMAZING!**👍🎆❤️

Looking directly at others’ work in order to copy parts of their code, taking credit for things you didn’t do
💔💥👎 **TERRIBLE! HORRENDOUS. PLEASE NO.**👎💥💔

Generative AI Policy

There is only one acceptable mode of use. The Prompt!

I have a question, but want to preface it by saying this is a question for a class I'm taking with a substantial programming component. I want to learn the material, so I only want you to answer conceptually to help me understand. Please do not provide any code examples longer than a single line, and err on the side of explanation about the underlying principles rather than simply providing an answer. If you are able to ask me follow up questions to help me figure it out by myself rather than directly explaining, even better. Can you do that?

The only acceptable GenAI / Chatbots / LLM use is:

Claude.ai (Sonnet 4.5), new chat per question, this *exact* prompt, AND sharing the public chat link in your assignment. (demo)

Generative AI Policy

ALL OTHER FORMS of GenAI use are 100% banned for work for this class… until the final project.

You need to learn a new way of thinking!

Heavy and generalized GenAI use will short-circuit this, leaving you unable to understand why things fail when they do.

Final project is more like the “real world” - anything goes! (just document what you did)

Generative AI Policy

Here’s the line:

Asking questions (using The Prompt) to help you understand concepts from class, like “how do I use a defaultdict?”
❤️🎆👍 **¡AMAZING!**👍🎆❤️

Asking LLMs to generate answers to copy-paste, providing sections of skeleton code to have LLMs fill in
💔💥👎**~~TERRIBLE. HORRENDOUS. PLEASE NO.~~**👎💥💔

Academic Integrity + GenAI Policy

Flexibility in structure does not mean flexibility on integrity.

Know that it can be very hard
to lie on a self-evaluation.

Cheating and dishonesty are so
unnecessary, and just so incredibly *lame*.

Don’t Waste Our Time,
Don’t Disappoint Dwight.

Agreements

I see this class as entering into a set of mutual agreements,
on top of the basic agreements of the university
(like academic integrity and so on)

We're building a community of learners interested in this topic!
(I'm a learner too.)

By registering, you agree to certain things -
By being the instructor, I agree to certain things.

You agree to:

Invest substantial time and effort in this course this quarter

Hold yourself accountable for your own progress

Be honest in assignments, self-evaluations

Stay on top of your work, and ask for help when needed

Be open to constructive feedback

Challenge yourself

Communicate with me when any of the above falls through

I agree to:

Invest substantial time and effort in your process of learning

Prepare well for class, construct meaningful assignments

Make myself available to help

Be open to criticism and commentary

Provide structures for learning

Communicate with you when any of the above falls through

The key idea for me is **mutual respect.**

I respect your time, intelligence, integrity, and effort.

I ask you to respect my time, intelligence, integrity, and effort.

WE ARE HERE TO HELP

No such thing as a dumb question here.

We're on the same team, this is not an adversarial relationship!

The Struggle!

Learning programming is like learning a new language

You have to soak in it and use it daily

It will feel unnatural at first, push through

Don’t be afraid to play around and break stuff

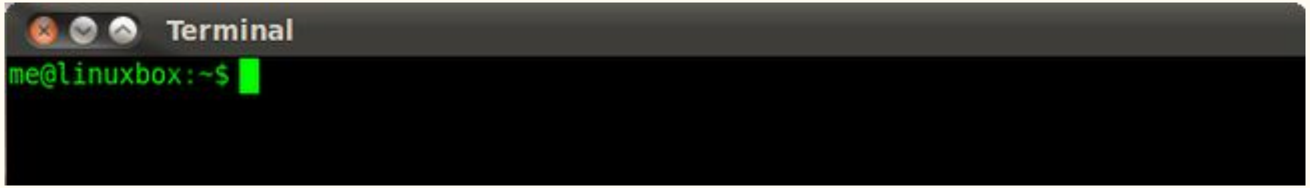
The Struggle Illustrated

YOU CAN DO IT

No such thing as a dumb question here.

ERRORS ARE YOUR NEW FRIENDS

Our new home: the Unix command line



```
me@linuxbox:~$
```

Precision - the challenge of exactitude

One wrong letter, space, or punctuation mark
can easily derail you

These mistakes are at first very hard to see

Double-check, triple-check your code
and relevant documentation
(a beloved acronym by programmers is RTFM - read the flippin' manual!)

Take a break and come back to it

Benefits of command line interfaces

Automatable

easy to do something 1000x

Fast

GUI interfaces are computationally ‘heavy’

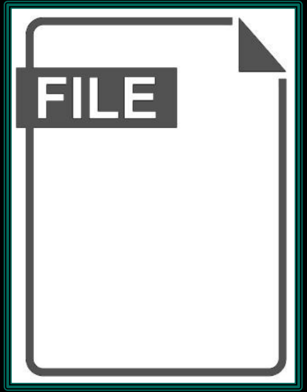
Consistent

same command always does the same thing

Transparent

you’ll learn what your files actually are

What is a file?



An abstraction!

... but ultimately, an array of bytes

e.g., for ASCII text:

Character	L	I	N	G
Bits	100 1100	100 1001	100 1110	100 0111

Types of Files

Text

bytes representing characters

txt, code (like .py), html, logs

Executable

compiled code in binary format

to run as a program

Data

everything else: images, zip files,

doc/ppt/pdf, and so on

file extensions are just a helpful suggestion!

Commands!

Usual format (space separated):

`program [flags] arguments`

`program`: Name of the command

`flags`: Often prepended with a -, these alter behavior of a command

`arguments`: These are information or locations the command needs

Use `man` to look up how to use a command, e.g.:

```
man cd
```

Will show documentation explaining the usage of the command.