

Welcome to COGS 18: Introduction to Python

COGS 18

Shannon E. Ellis

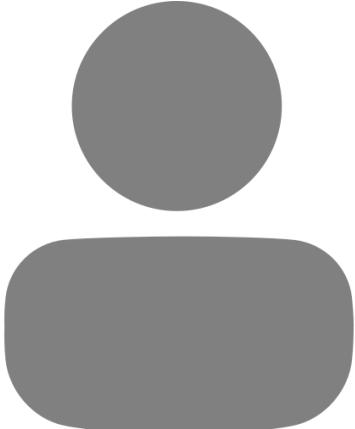
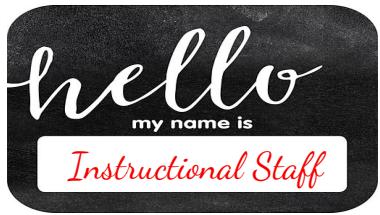
Associate Teaching Professor

Department of Cognitive Science, HDSI

✉ sellis@ucsd.edu

○ ○ ○

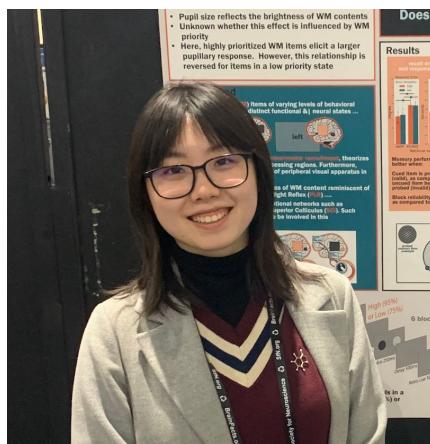




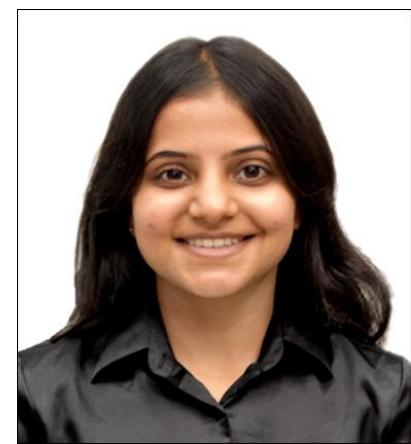
Prasannakumaran



James



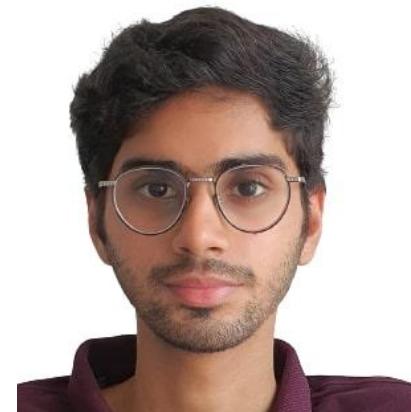
Yueying



Saanya



Kunal



Samyak

TAs

IAS



Doris



Katie



Laura



Nian-Nian



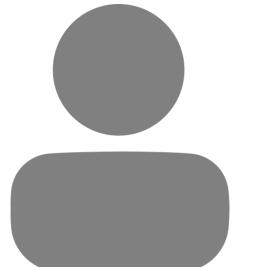
Sandy



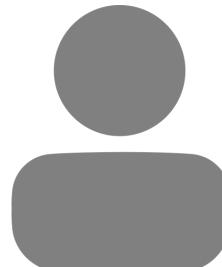
Ana



Abhay



Margaret



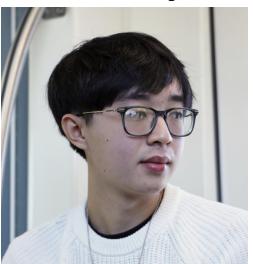
Sophie



Liz



Keshav



Nick



Eric



Ashesh



Nikita

The (dreaded) waitlist

1. I do not handle the waitlist - our staff (cogsadvising@ucsd.edu) do
2. I do not have access to the waitlist nor the system that enrolls students from the waitlist.
3. Typically ~3-5 students from each section are enrolled by our staff.
4. Enrollment cannot exceed no. of seats in the classroom.
5. The waitlist clears at the end of week 2.
6. You should complete the pre-course assessment if you hope to be enrolled.

If you email me about the waitlist or your specific circumstance/need to take this course this quarter, I will point you to cogsadvising@ucsd.edu.

Let's chat: Teaching &
Learning Programming

o o o

Why Python?

simple(r) syntax

widely-used

Jupyter Notebooks

*"It's not the best language for anything, but
it's the second best for everything"*



Intro Programming courses
are often **thought of as**
difficult and are courses with
the highest dropout rates

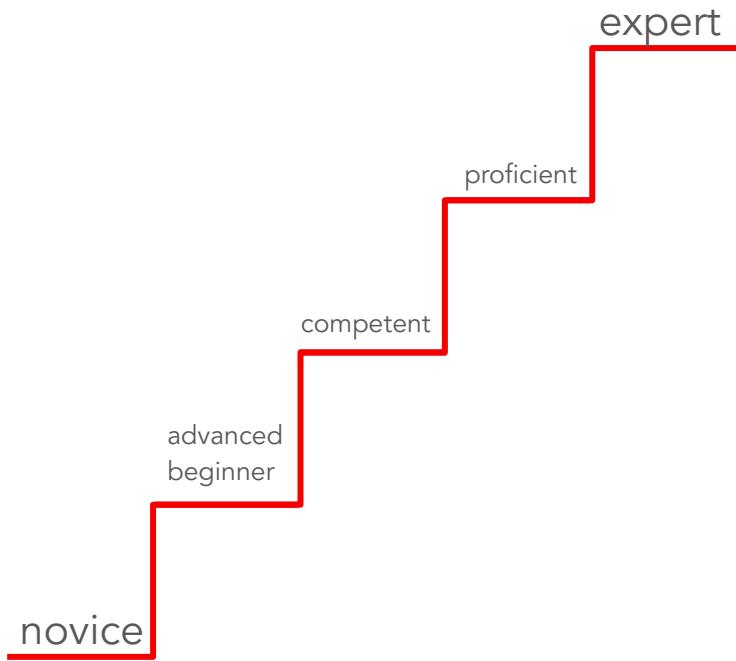


```
31     self._init()
32     self.file = None
33     self.fingerprints = set()
34     self.logdups = True
35     self.debug = debug
36     self.logger = logging.getLogger(__name__)
37     if path:
38         self.file = open(os.path.join(path, "fingerprint"))
39         self.file.seek(0)
40         self.fingerprints.update([line.strip() for line in self.file])
41
42     @classmethod
43     def from_settings(cls, settings):
44         debug = settings.getbool("DEBUG", False)
45         return cls(job_dir(settings), debug)
46
47     def request_seen(self, request):
48         fp = self.request_fingerprint(request)
49         if fp in self.fingerprints:
50             return True
51         self.fingerprints.add(fp)
52         if self.file:
53             self.file.write(fp + "\n")
```

....yet, the only thing that is slightly predictive of success in an intro programming course is...*how successful the student thinks they will be*

Things that do NOT predict success:

- gender
- age
- personality
- math ability



My goal is to have you all be able to **program at an introductory level**

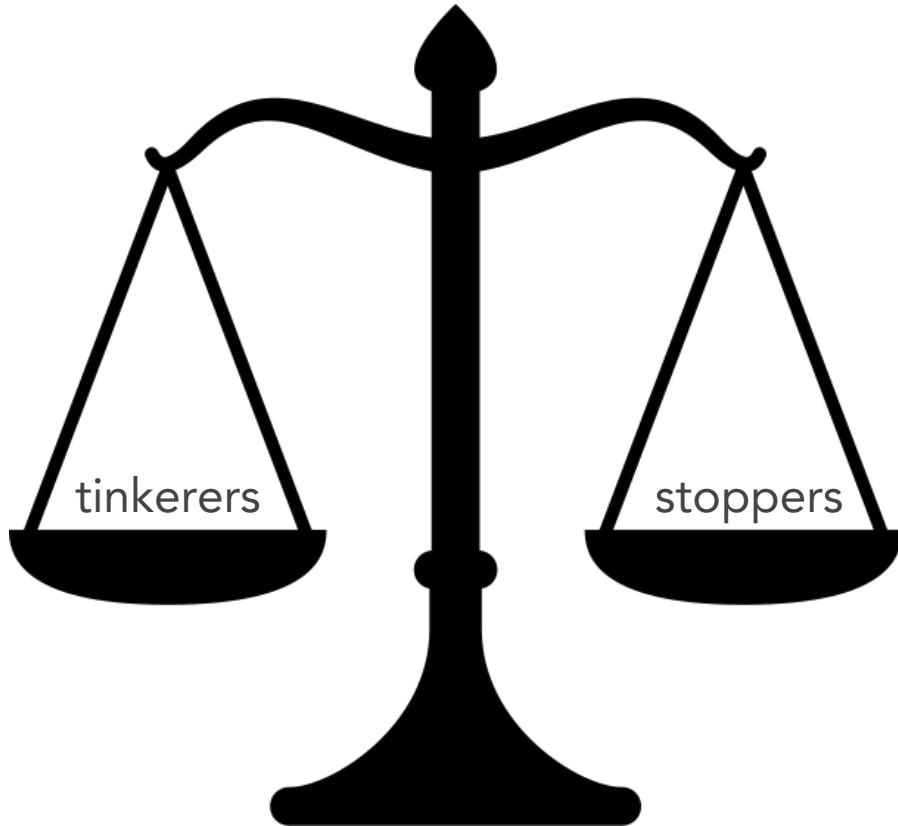
It's generally accepted that it takes people **10 years to move from novice to expert programmer**. But, there are lots of steps in between! We're working to move you further away from novice (& in the direction of expert) than you are right now.



Mixed Messages: We tell people learning to program will be **tough and frustrating** but that **if you're not having fun, you're doing it wrong**.



Building Blocks: Too often, we also tell people to “just try things out” without explaining basic concepts. Other courses aren’t taught this way...



Be a mover: Make forward progress. Strike a balance between just stopping and tinkering forever.

If you're not moving forward, consider the **2-hour rule**.

If you're trying to figure something out and struggling to move forward at all, consider the 2-hour rule. If you're stuck, **work on the problem for an hour**. If you're still stuck, walk away & **take a 30 min break**. Then, **try again for another 30 minutes** or so. If you're **still completely stuck, stop and contact us** (come to office hours, post on Piazza). If you're not even sure what your question is, include what information that you do have - what you're stuck on, what you've tried, error messages you've received, etc.



COGS 18: How this course is going to work

o o o

To avoid the common pitfalls of intro programming courses, we're going to take the following approach:

Note: iclickers are for *extra credit only*; every lecture attended is EC to the final (up to 5% on final); can attend either lecture time on a given day

1. First 2/3 of course: **basic concepts**
2. **In-class practice** (no stakes)
 - a. iclicker for comprehension
 - b. time to apply what was just explained
3. **Coding Labs** (low stakes)
 - a. Notebooks provided
 - b. Staff/classmates there to help
 - c. Checked for effort, not correctness
4. **Assignments** (mid stakes)
 - a. Completed individually (*can work together*)
 - b. Programmatically graded
5. **Midterms** (high stakes)
 - a. Two parts: conceptual (in-class) + technical (take-home)
 - b. Completed totally individually

COGS 18: How You'll Be Evaluated

	% of Grade	Requirement
Pre- and Post-Assessment	4%	Complete
Coding Labs	16%	Participate In 8 Coding Labs
Assignments	30%	Complete 5 assignments
Midterms	30%	2 midterms
Final	16%	Complete final project OR final exam

The “I already know Python” option

Graded on:

- Pre- and Post (2% each)
- Midterm Exams: both in-class and take-home (25% each)
- Final Exam OR Final Project (46%)

Opt in:

- Via [Google Form](#) (link also on syllabus)
- Once you opt in you cannot change your mind
- If you are unsure, proceed with full course for now; you can always opt-in later.

Pre- and Post-Assessment surveys (4%)

Students will have to complete two surveys at the beginning of the course: 1) Coding pre-assessment (~30 min); 2) Information about you (~15 min)

Notes:

1. Be sure to **type your PID in correctly** to the survey. This is being used to determine you've submitted both surveys.
2. This is an intro programming course. **You are NOT expected to know the answers to the questions on the pre-assessment yet.**
3. Completion is required for this course (#finaid); *however*, YOU get to decide if you opt into your responses being used for research. Also, correctness does not matter for credit, but real effort does.

CodingLabs:
apply concepts
discussed in
lecture using
coding labs
(16%). Practice
makes progress.

Attempt for full credit (2% each)

- Have to make a concerted effort to complete **labs**
- Coding Labs will be submitted on datahub
- Answers will be sent out the following week
- Encouraged to work with others

You **should attend the section to which you're assigned.** You *can* attend a different section. However, if one section becomes too crowded each week, we'll revisit this policy.

Note: CodingLabs (and Office Hours) begin **Week 2**

o o o

**(5) Assignments
(30%)** : Jupyter notebooks that are completed individually & graded programmatically.

Assignments due Tues @ 11:59 PM.

Assignment	Week	Median Time Spent (hours)
A1	wk3	2
A2	wk4	4
A3	wk6	4
A4	wk7	5
A5	wk10	5

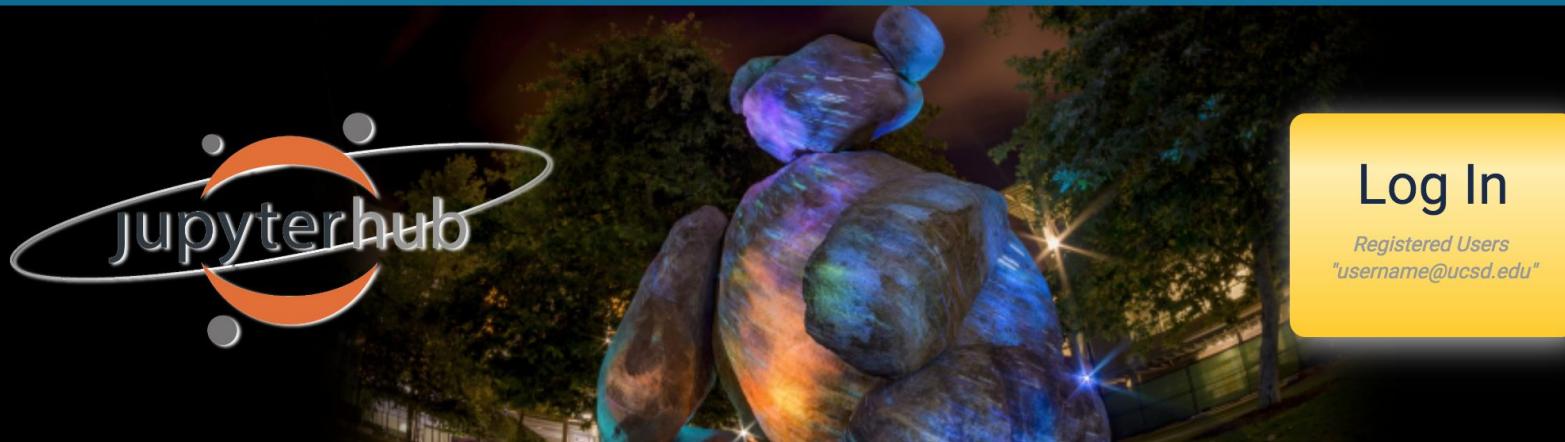
Assignment Submission @ Datahub: <https://datahub.ucsd.edu>

DATA SCIENCE / MACHINE LEARNING PLATFORM

UC San Diego

Information Technology Services - Educational Technology Services

Help Options ▾



UC San Diego Jupyterhub (Data Science) Platform

o o o

In technical
classes, Piazza is
a particularly
helpful resource

Please don't send me a
Canvas message. The UI is
the worst and I miss
messages and then feel bad.

Order I reply:

1. Piazza
2. Email
- ~~3. Canvas~~

There are rules:

1. No duplicates.
2. Include Assignment & Question in Summary line.
3. Posts must include your question, what you've tried so far, and resources used.
4. Public posts are best.
5. Helping one another is encouraged.
6. No assignment code in public posts.
7. We're not robots.

(2) Midterms (30%):
will be completed
individually.

Two parts:

- In-person: conceptual
- Take-home: technical
(open-book/open
Google/open ChatGPT)

Each part will be **completed on
your own**. These will include a
combination of types of questions.

(1) Final Project or Exam (20%): will be completed individually and submitted electronically on the day of the final.

It will be up to you which you do. The **project** will help you learn more and has the opportunity for EC and an A+ in the course, but takes longer. The **take-home exam** takes less time, but the highest grade you can earn in the course is an A and must be completed on your own. You do not have to show up anywhere on the day of the actual final.

All exam and due dates are all listed on the schedule on the course syllabus and are in Canvas



Your point of contact for COGS 18
will be the course website:
<https://cogs18.github.io>

Course Website	<u>https://cogs18.github.io</u>	syllabus, Coding Lab Answers (& lecture notes)
Piazza	<u>https://piazza.com/class/luagsvdfwro6ek</u>	questions, discussion, regrade requests
Canvas	<u>https://canvas.ucsd.edu/courses/54990</u>	grades, OH times/locations
Datahub	<u>https://datahub.ucsd.edu/</u>	coding labs, assignments, exams, (& lecture notes)
iclicker	<u>iClicker Student App</u> OR <u>Instructions to register remote</u>	Need account with <i>iclicker</i> using ucsd email (sign-in if you have one)
Anonymous Feedback	<u>Submit via Google Form</u>	if I ever offend you, use an example you hate, or to provide general feedback

Any questions about
course logistics?

• • •



Where to turn for help
and practice when
learning to program?

o o o

Many Avenues for help

- Office Hours
- Classmates
- Piazza
- AI/LLMs (ChatGPT, CoPilot, etc.)
- Google with “in python”
- StackOverflow
- GitHub

You **ARE** encouraged to help one another in COGS 18.

ChatGPT

ChatGPT



Examples

"Explain quantum computing in simple terms" →

"Got any creative ideas for a 10 year old's birthday?" →

"How do I make an HTTP request in Javascript?" →



Capabilities

Remembers what user said earlier in the conversation

Allows user to provide follow-up corrections

Trained to decline inappropriate requests



Limitations

May occasionally generate incorrect information

May occasionally produce harmful instructions or biased content

Limited knowledge of world and events after 2021

Send a message...



When **CAN'T** you use LLMs in COGS 18? In-class Exams

When **CAN** you use them? Everything else.

...but you need to be able to explain everything you turn in.

Be able to answer yes to BOTH:

- Can I explain each piece of code and each analysis carried out in what I'm submitting?
- Could I reproduce this code/analysis on my own?

Using AI/LLMs as a beginner

Process:

- Try on your own first
- Avoid copy + pasting whole questions
- Come up with your own questions
- Read *and think* about the answers
- Work to understand

How we'll help

(and how to best help one another)

Dialogue-based | questions & back-and-forth

Student-driven | work to help you find the solution

Example-heavy | ask about other examples, refer back to previous work, etc.

Scaffold with prior knowledge | review material first to get prerequisite knowledge

A message for
first-gen students,
transfer students,
and those who
don't have older
siblings/friends who
have attended
UCSD/university

If you are struggling, come to office hours. Ask questions on Piazza. Reach out to me to ask for better approaches. Your classmates ARE doing this. And, you're not alone.

If you need a bit longer on something b/c you fell sick, a family thing came up, work called you in for an extra shift, etc., ask for an extension. Your classmates ARE doing this.

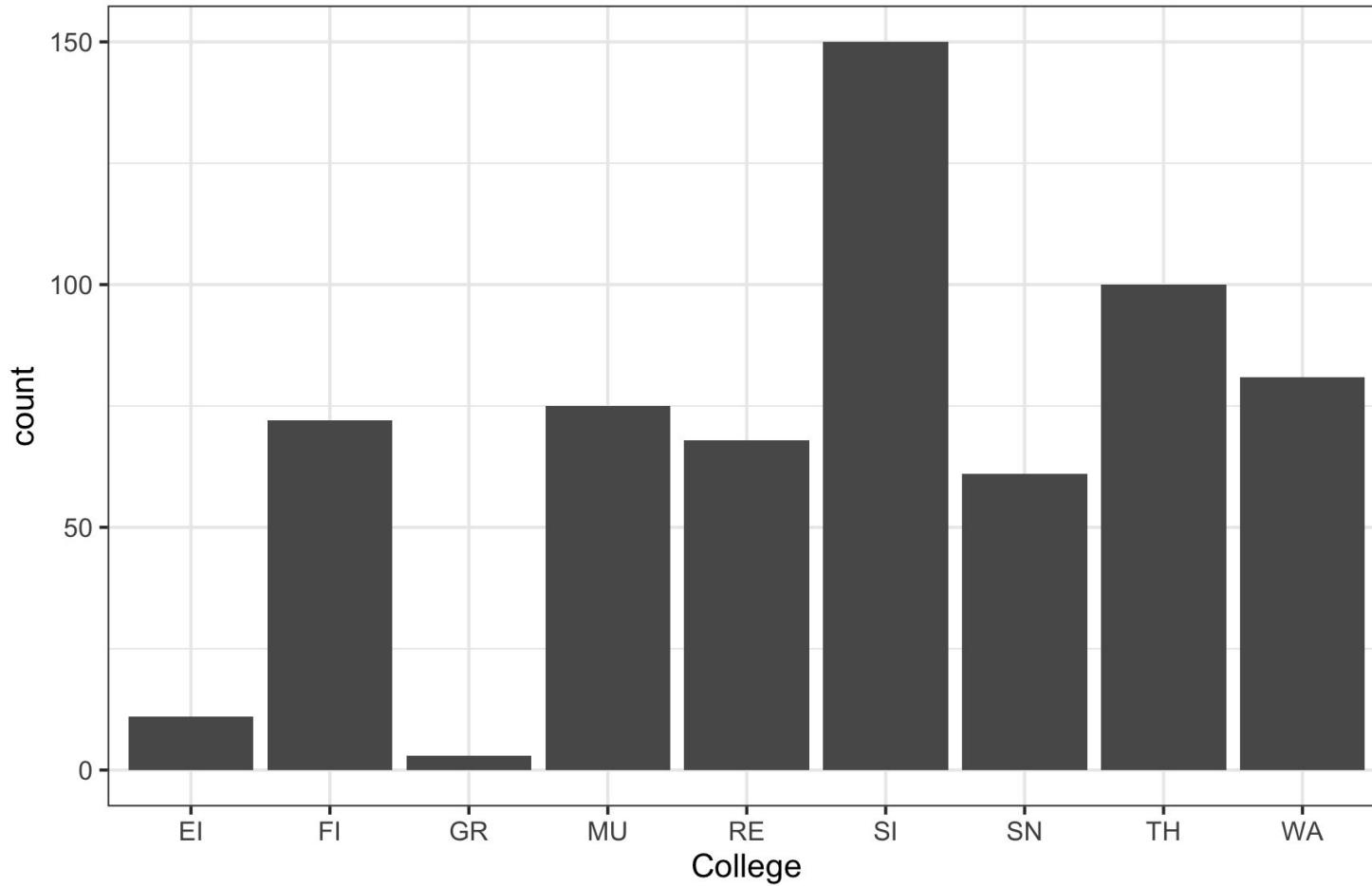


Today I used a PDF slideshow,
but every other day of class,
lecture notes will be presented
in a Jupyter notebook

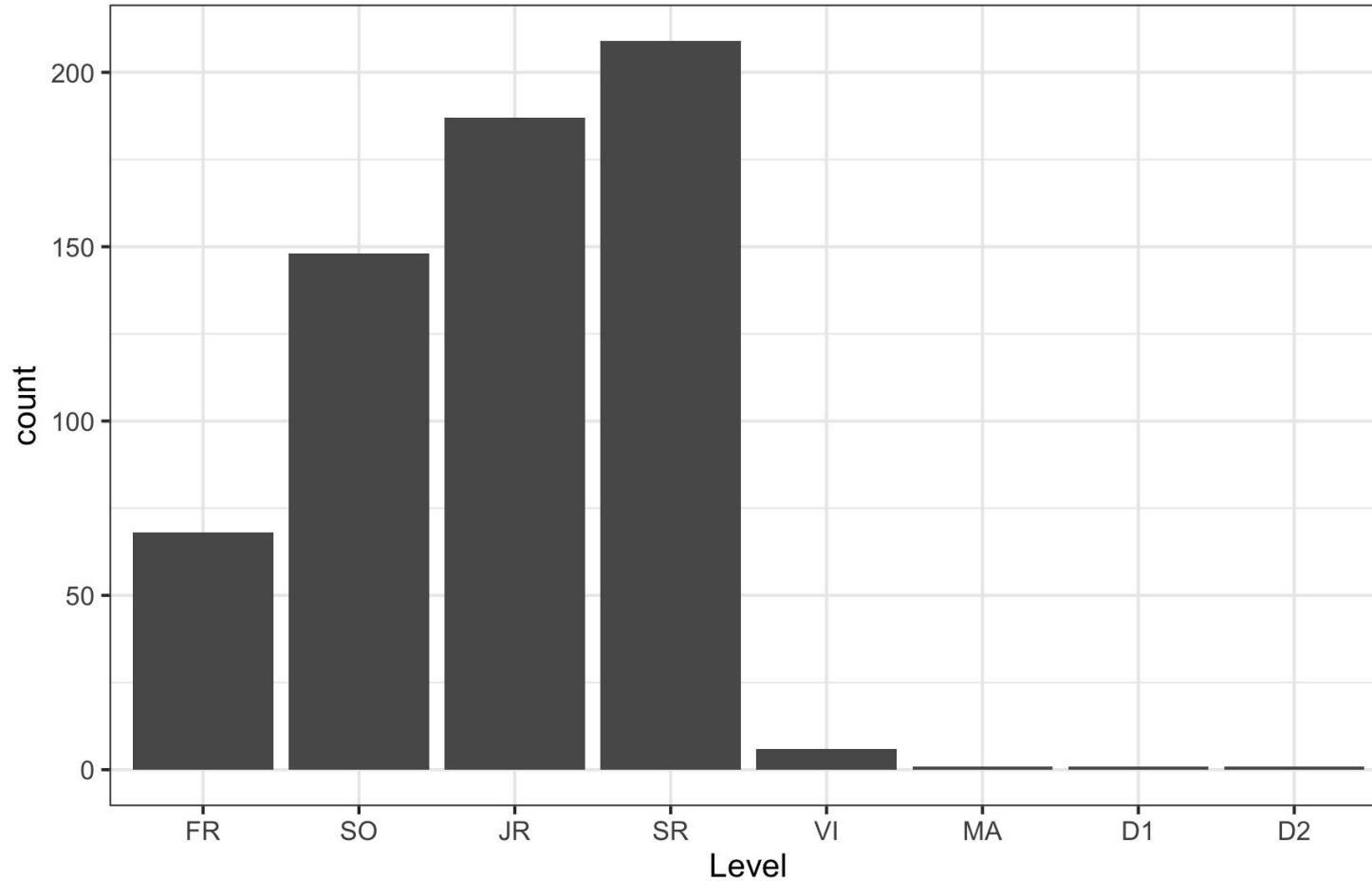
ooo



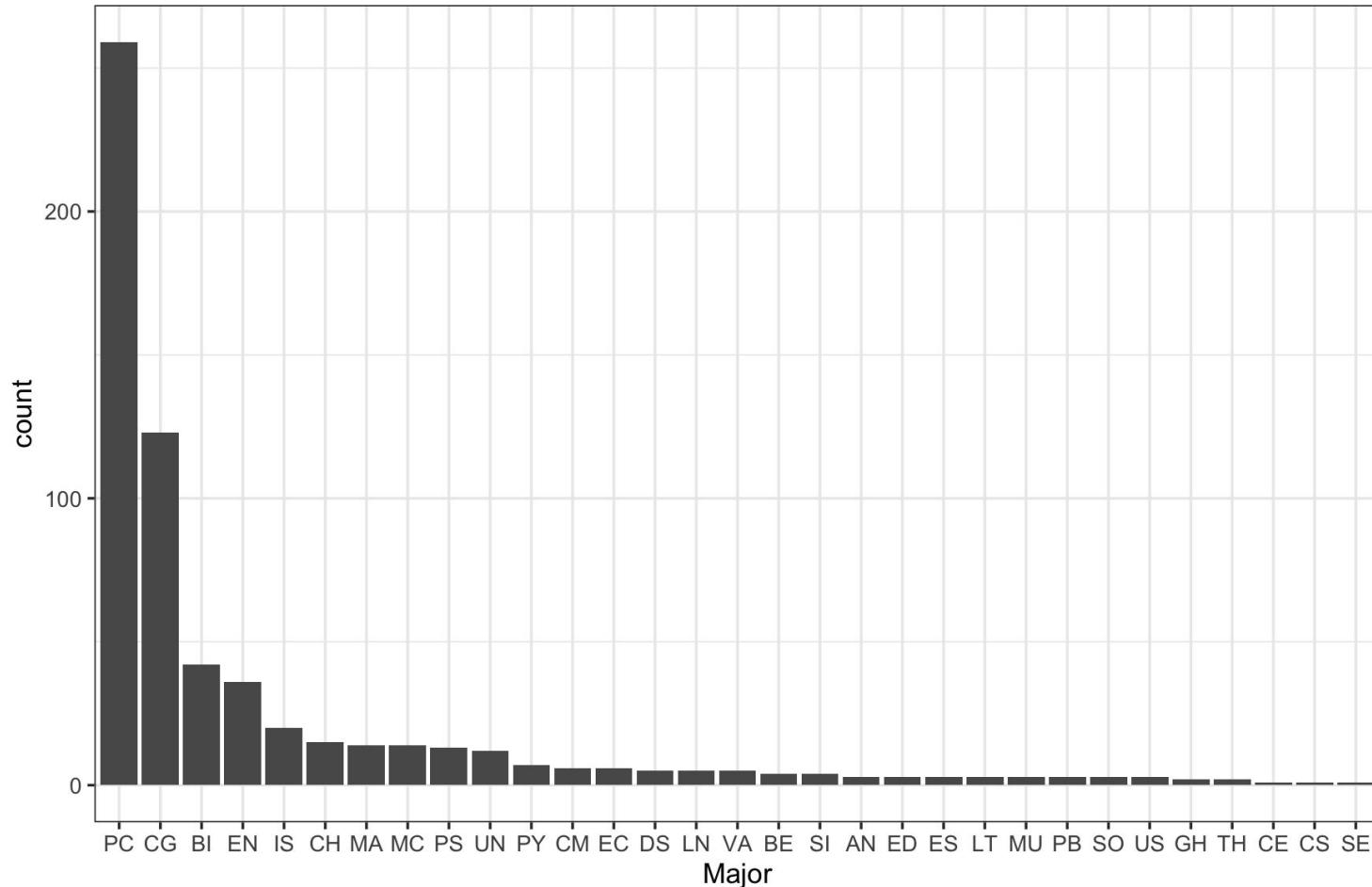
COGS 18: Colleges



COGS 18: Year in School



COGS 18: Majors





Please complete the two pre-assessment surveys before next Tuesday at 11:59 PM.