# Discussion 13

## Final Review

Mingxiao Wei
mingxiaowei@berkeley.edu

April 20, 2023

# From last time ... 👀

- eecs16a
- ee16a
- UGBA 13
- data8
- Data8
- Music 26AC, Music in America! - wow I'm actually taking this now
- So far 61A - ME TOO!
- physics 7c
- Mingxiao's CS61a sections - awww <3
- Data 8
- BIOLOGY 1B - in person labs must be fun but I took in during the pandemic rip
- cs61a
- econ 100a
- Data 8

"What is your favorite course taken at Cal?"

- German C25
- Math 1B
- DATA 8
- Organic Chemistry - 👀who's ur ochem prof? I took it w Prof Robak and she was SO nice
- Data 8
- BIOE171 - "Interface Between Neuroethology & Neural Engineering" <- omg that sounds so interesting
- data 8
- 61a :)
- CS61A
- cs61B
- CS61A~
- Astro C10
- Math 55

# From last time … 👀

- What is it like studying CS and MCB at the same time? How stressful is it?
  - A lot of classes to take (esp. mcb lower divs) - if you are in a similar situation I'd recommend using your high school exams to fulfil requirements if possible
  - Other than that, it is fun! I like both subjects, and it's exciting to learn knowledge in one field that can be applied in another/explore the intersection of the two
- Any advice for studying for finals? What is your favorite restaurant near Berk
  - Finals - see the next two slides
  - Sushi Secret 💯
- I miss the slide pdfs, along with google drive link do you think you could upload pdf of slides?
  - You can download the pdf from google slides by navigating to Files → Download → PDF but I have also uploaded them to our website!

# Logistics

- `(scheme)` 👀
  - The entire project due tomorrow 04/28
  - Submit everything today for 1 extra credit!
  - Go to OH/project parties!
- Homework 10 due next Tue 05/02
- Homework 11 due next Thu 05/04
  - Not released yet; just be aware that there's another homework
  - No actual questions, just a bunch of surveys/course eval/Scheme contest voting
- Reminder about homework 09 recovery (Ed #3083)
- This is officially our very last section □

# About the final □

- 11:30AM - 2:30PM on Tue 5/9
- If you need any accommodations (remote/left-handed desk/alternate time/DSP/etc.), fill out the form go.cs61a.org/exam-alts by next Mon 5/1!!
- Check out Ed #3122 for more info on logistics/resources!
- TIPS
  - Make sure you are familiar with all topics in scope
    - Review sessions are great for this! If you don't have time to attend/watch recordings, going over their slides will be helpful too
  - familiarize yourself with the study guide too! (linked on cs61a.org/)
  - If not confident enough to start doing exam questions, review discussion worksheets first (which have walkthrough videos!)
  - If you want practice by topic: https://cs61a.org/resources/
  - **Do past exams**, ideally 3-4, or more, but quality > quantity. Some has walkthrough videos too
    - Start slow; allow yourself enough time to think about each problem (but not too long)
    - Time yourself for the last 1-2 exams you do

# About the final □

- Other exam taking tips
  - **<u>READ THE PROBLEM STATEMENT</u>**
  - **<u>READ THE DOCTESTS</u>**
  - Run your code as if you were a Python interpreter and see if it produces the desired output
- I'll send out a doc with more problem solving tips for each topic this weekend/early in RRR week!

# Today’s outline

From your responses in the lab checkoff form:

- Trees - Q4
- Recursion - Q1
- Generators - Q6
- Scheme - Q7
- SQL - Q8, 9, 10
- Linked lists - Q5
- List mutation - Q2

# Recursion/Tree Recursion

1. Read the problem statement. Understand what the function does conceptually
2. Read at least the first two doctests.
   a. Does the output make sense to you?
   b. If it's a tree, draw it out!
3. Base case
   a. Read the rest of the doctests - the base case may be hidden in one of them!
   b. If not sure, leave it first and work on the recursive case. After you finish the recursive case, think about how the arguments change throughout the recursive calls
4. Recursive case
   a. If the input is a list (including Scheme lists), most of the time we handle the first element in the list, and the recursive call handles the rest
   b. Use one example from the doctest, think about what the recursive call should be for this input - i/e., how to break down the problem into a smaller one. Then generalize.

# SQL

1. What columns does each table have?
2. What table contains the info needed to get the output?
   a. If > 1 table, join them together → Cartesian product. How can we filter out unuseful rows?
   b. If we need to make > 1 pass through a table, join the table with itself
3. `GROUP BY` only makes sense with aggregation
   a. `SELECT ▲ FROM … GROUP BY [col] …` → the columns in ▲ must either be `[col]` or a call to an aggregation function
      i. Aggregation function is applied once to each group
      ii. Each group turns into one row in the output
      iii. Rows in same group may have different values in columns other than `[col]`
   b. `HAVING` - filter on groups; `WHERE` - filter on rows

# That’s all folks!
# Thanks for a great semester
# & best of luck ❤️

go.cs61a.org/mingxiao-att