# Welcome to 61A Lab!

We will begin at **5:10**!

Slides: **cs61a.bencuan.me**

# Announcements

- This is the last lab :')
  - Final discussion this Thursday!
  - Topical review sessions next week
- HW8 due Thursday
- Scheme due tonight
- Scheme art contest due tomorrow

# The Plan

- Final overview
- Ask us anything!
- Exam problem walkthrough
- Requested topics

# Final Info

# Logistics

- not out yet - see piazza!!
- will be on Tuesday 5/10 from 11:30-2:30
- 75 course points with typical average of ~45-50
- same policies and format as midterms
  - 3 two-sided cheat sheets
- hopefully less time crunchy than midterms

# Stuff to study

- All pre-MT2 content (large portion of final)
  - Control, higher order functions, lambdas
  - Tree recursion
  - Trees
  - OOP, inheritance
  - Linked lists
  - Iterators (and generators - less emphasized)
  - Efficiency (less emphasized)
  - String representation (less emphasized)

# Stuff to study pt. 2

- Post MT2 content
  - Interpreters: eval/apply mutual recursion, lexical vs syntactic analysis, Scheme project design
  - Scheme: basic syntax, lists, data abstraction, programs as data
  - Regex: basic expressions using re.search()
  - BNF: read and write basic grammar, read and draw syntax trees based on input

**Refer to piazza for exact scope!**

# Useful Resources

Also posted on cs61a.bencuan.me!

Ben's midterm studying strategy guide:
https://cs61a.bencuan.me/Midterm-Tips-sp22-858964ddc43343cea52f6afbb2af05cf

Tanay's list of useful midterms and problems:
https://sparkling-swamp-b74.notion.site/CS-61A-Resource-Guide-6c4b98c53089424f9554fff9b1107698

CSM study materials:
https://docs.google.com/document/d/145kJlPtrbu410SYVhyznOV19gPCm5EOZ6XbtReOk33o/edit

# Tips from the AI’s

- Thoroughly look over solutions for MT1 and MT2 to identify how/why you went wrong so you don’t make those same mistakes on the final
- Get lots of sleep + take care of yourself! It’s okay to stop studying when you feel ready, you don’t have to keep going because everyone else seems to be

# Lab

Hints available upon request, but try to get as far as you can on your own!

Good optional problems to start with:

- Q6 (mutability, trees, tree recursion)
- Q13 (difficult linked lists problem)

**(b) (6.0 points)**

The `join` procedure takes two lists of lists `s` and `t`. It returns a list of lists that has one element for each possible pairing of an element of `s` with an element of `t`. Each element of the result is a list that has all the elements of a list from `s` followed by all the elements of a list from `t`.

For example:

```
scm> (define instructors '(
       (john 61a)
       (hany 61a)
       (josh 61b)))
instructors
scm> (define grades '(
       (a b)
       (c d)))
grades
scm> (join instructors grades)
((john 61a a b) (john 61a c d) (hany 61a a b) (hany 61a c d) (josh 61b a b) (josh 61b c d))
```

Implement `join`.

```
(define (join s t)
  (if (null? s) nil
      (append (map (lambda (v) (append (car s) v)) t)
              ;  (a)      (b)  (c)      (d)  (e)
              (join (cdr s) t))))
              ;             (f)
```