Welcome to CS61A Lab 3 Sect. 29/47 :D

Dickson Tsai
dickson.tsai+cs61a@berkeley.edu
OH: Tu, Th 4-5pm 411 Soda

Previous: Control Flow and HOF **>>>**
Today: Lambdas and Recursion >>>
Next stop: Midterm Post-Mortem

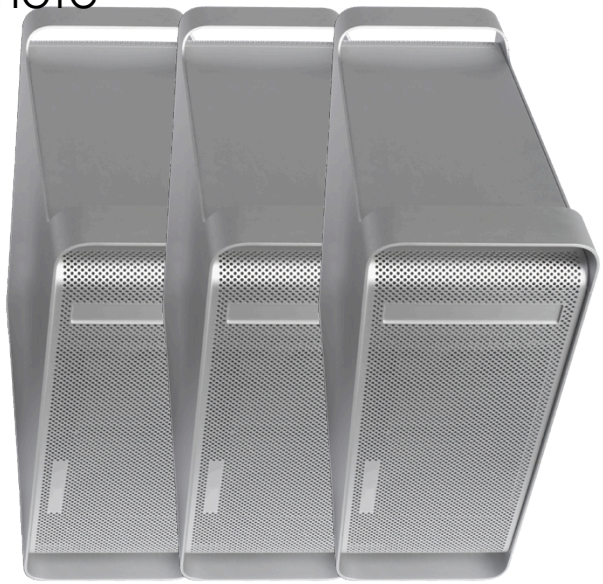
Section Quiz Policy

- New policy
  - In-class quizzes graded on effort to mark discussion attendance
  - Effort = Correct your answers using a different color
  - Please still be prepared for discussion.
- My new e-mail: dickson.tsai+cs61a@berkeley.edu

Submission Process Review

`ssh cs61a-??@cory.eecs.berkeley.edu`
connect to remote terminal

`scp ~/path/to/file.py cs61a-??@cory.eecs.berkeley.edu:~/dest/folder`
Copy file to remote server



`submit lab03`

Actually give your files to us!

- Make sure you are in the right folder
- Do not include .py ending here
- Do not submit from ~ folder

`cory[2] ~ #`
Your prompt should now say that you are in Berkeley's server! UNIX is used here!!!

Anuncios

- Project 1 due 11:59 PM 9/17 (Wed) for 1 pt extra credit
  - Due 11:59PM 9/18 (Thr) for normal credit
- Midterm 1 next Monday 9/22.
  - Practice problems until you have strong + correct intuition
- Reminder: My OH are Tuesdays, Thursdays 4-5 PM in Soda 411

Tip of the Day

- Iterative Improvement!
  - Do little by little so you won't be overwhelmed by the end of the week
  - The down time between classes/at meals is a great time to sneak in an env diagram or implementation of factorial
  - Try the easiest problems first. E.g. do the play function in Hog after enough experience with simple iteration Qs.
  - The more you do, the faster you'll be at them, and the less tedious they become.
    - E.g. first learning the bicycle
      - Start slow. Do not bike up the hill next to Soda. Somewhere flat.
      - Get more comfortable w/ practice
      - Now you can go to far-away places and enjoy the sights!

Directions

1. Access the lab at cs61a.org
2. Type the WWPP question into your interpreter. Now try to explain its behavior to your neighbor. You guys can also work on drawing an env diagram together.
3. Must submit: Q3, Q4, Q6, Q8 in `lab03.py`

To get to my website

- dicksontsai.com/cs61a
- OR
  - Go to cs61a.org
  - Click the “Staff” link
  - Click my name
  - Go to Discussion for discussion materials

- You will find my slides to discussion/lab. They will be different, because I add more content along the day based on student feedback.

Optional Conceptual Questions (Friday 9/12 lecture)

1. Notice that improve is a solution to an iteration problem. Try applying the iteration questions to it and see how the answers play out in the code: What to repeat? What to keep track of? Which value to start/stop with for each var? What to do w/ results?
2. What is the domain/range/behavior of the improve function?
3. If I wanted to find a zero of $x^2 + 2x + 1$, how will I do it with the code presented in lecture?
    1. How would you represent the mathematical function $f(x) = x^2 + 2x + 1$ in Python?
4. What is the inverse function of $f(x) = x^2$? How do I define it with Newton's Method?
5. Same question worded differently: What is the purpose of Newton's Method? What is it used for? What is its behavior?

Optional Conceptual Questions (Monday 9/15 lecture)

1. What are the components of a recursive function?
2. How should base cases be checked?
3. Why should base cases not be evaluated with recursive calls?
4. Iteration is a special case of recursion. Presumably, we can use the same questions from iteration on recursion problems. Try it on factorial.
    1. What computation do we want to repeat?
    2. What variables do we have to keep track of?
    3. What values do the vars start with?
    4. When should the computation end? What do we do with resulting values?
5. How does the environment diagram of recursive factorial compare to iterative factorial? How do both functions keep track of their current state?
6. Why is iteration a special case of recursion? What is either good for?