Welcome to CS61A Lab 2 Sect. 29/47 :D

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

Previous: Your Own Machine **>>>**
Today: Control Flow and HOF >>>
Next stop: Lambdas and Recursion

Optional Section Homework [Canceled]

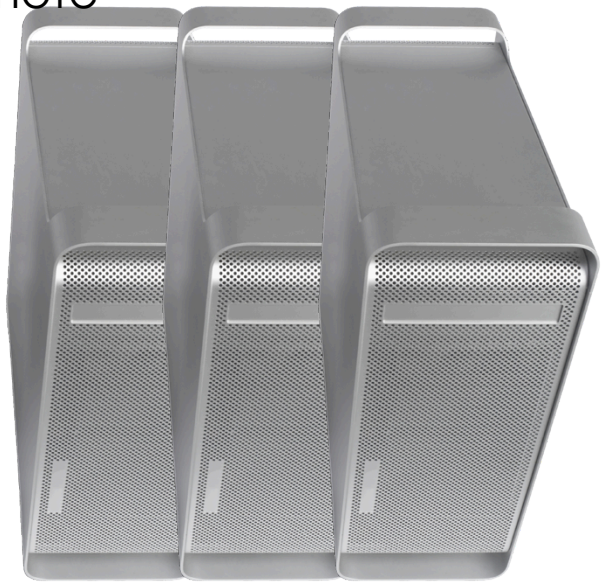
- This will be used to mark attendance for the next section for midterm recovery. Will help participation (but you can participate in other ways as well!)
- These should take at most 15 minutes, if you keep up with your reading.
- Submit with voice recording, email, paper, slideshow, however you want.
- Graded on effort, but use your time wisely. 10 minutes of BS < 10 minutes of ***citing*** the textbook
- Feel free to make your own questions!

- New policy
  - In-class quizzes graded on effort to mark discussion attendance
  - Please still be prepared for discussion.
  - Those who submitted this assignment by Thursday 9/4 will have a free absence from discussion for midterm recovery.

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 lab02`

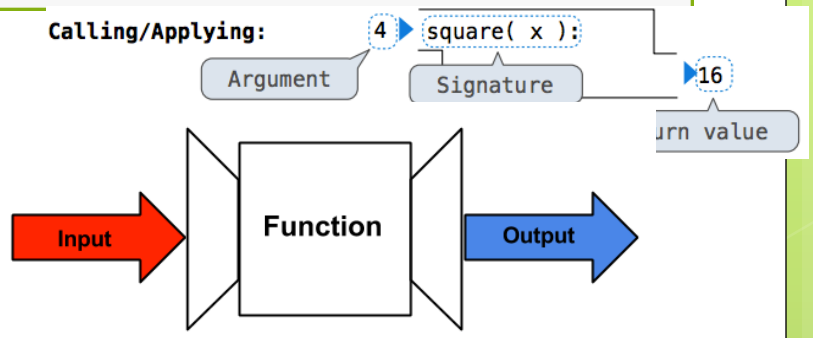
Actually give your files to us!

- 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

- HW1 due 2pm (before lecture) on Wednesday 9/10
- Project 1 due 11:59 PM next Wednesday 9/17
  - START NOW!!!
- Take-home quiz posted 3pm this Wednesday, due 11:59PM this Thursday 9/11
  - To see if you are on pace with the course
- Reminder: My OH are Tuesdays, Thursdays 4-5 PM in Soda 411



Tip of the Day

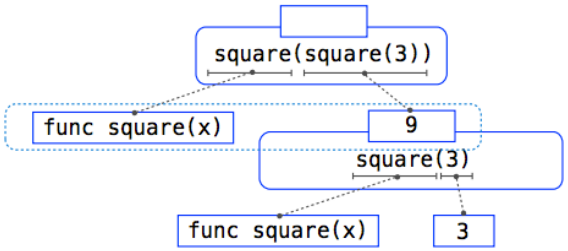
- Visualize everything!

Control flow: think of a river! Photo cred: Wikipedia user Qaalvin

Function:

- List what you know/assume about the input/output.
- Summarize what the function does (behavior) in the middle box!
- Photo cred: Simon Ko

Call expressions:



Tip of the Day

- Visualize everything!
  - Environment Diagrams!!!
  - Work with Python Tutor
  - But remember that PythonTutor does not show everything.
    - In particular, you have to keep track of where you are in the rules (e.g. I'm still evaluating the operator of this call exp)

Tip of the Day

- Visualize everything!
  - Higher order functions:
  - E.g. factory that makes cars (customize functions)

```
def make_car(color):
 def car(driver):
  print(“‘Vroom! I like this {0} \
   car!’ -{1}”.format(color, driver))
 return car
make_car(‘blue’)(‘Ray’)
my_car = make_car(‘green’)
my_car(“dickson”)
```

Car factory. Photo cred: Brian Snelson

Q: How can we make sure we don’t mix the colors of two different cars together?

Directions

1. Write your name and login on the board (or email to me your login, e.g. cs61a-ty) if you went to discussion on Thursday. I will cross-reference your names with my attendance sheet, so please be honest.
2. Access the lab at cs61a.org
3. Must submit: Q3, Q7, Q9, Q12 in `lab02.py`
4. Conceptual questions: Q1, Q2, Q6, Q8, Q11
5. Extra coding practice: Q4, Q5, Q10, Q13 in `lab02_extra.py`
6. ---Loop--- If you are stuck, please collaborate! (even just discussing the WWPP problems)

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 may be different, because I add more content along the day.

Optional Conceptual Questions (Friday 9/5 lecture)

1. Creating a new frame and binding the formal parameters to the arguments happen in the (circle one:) eval / apply phase.
2. State the rules for executing a **`def`** statement. Why is nothing in the function executed until the function gets called?
3. State the rules for evaluating a name (name lookup).
    1. Why does each function call have its separate environment?
    2. True/False: we always look at the frame above the current frame if we cannot find the name in the current frame.
4. What would Python output? (→)
5. Before we enter the while loop, do we have to check the condition header? When might this be useful?

```
def foo(x):
  if x > 1:
    print(1)
  elif x:
    print(2)
  else:
    print(3)
print(foo(0), foo(1), foo(2))
```

Optional Conceptual Questions (Friday 9/5 lecture - Python)

1. Write an assignment statement to bind names `a, b, c, d` to the output of `square_four`.

```
def square_four(a, b, c, d):
    """Takes in 4 numbers and returns each number's square.
       Returns 4 numbers in total."""
```

2. Write `doctests` for `square_four.`
3. Change the func signature below to make `best` default to `True`.

```
def go(bears, best):
```

1. What part of the eval/apply rules for evaluating call expressions do we have to modify to support default arguments?

Optional Conceptual Questions (Monday 9/8 lecture)

1. Explain how the following line works:

```
assert x == 2, "x is not 2"
```

1. The characteristics of a function are the domain, range, and behavior.
   1. Define each of these terms.
   2. Find the domain, range, and behavior of this code:

```
def pyramid(n):
  a, b, total = 0, n, 0
  while b:
    a, b = a+1, b-1
    total = total + a + b
  return total
```

2. What're advantages of higher order functions (HOFs)? Disadvantages?
3. Give examples (real life examples ok!) of how: HOFs express general methods of computation, HOFs remove repetition from programs, HOFs separate concerns among functios.