Control Structures

David E. Culler

CS8 – Computational Structures in Data Science

http://inst.eecs.berkeley.edu/~cs88

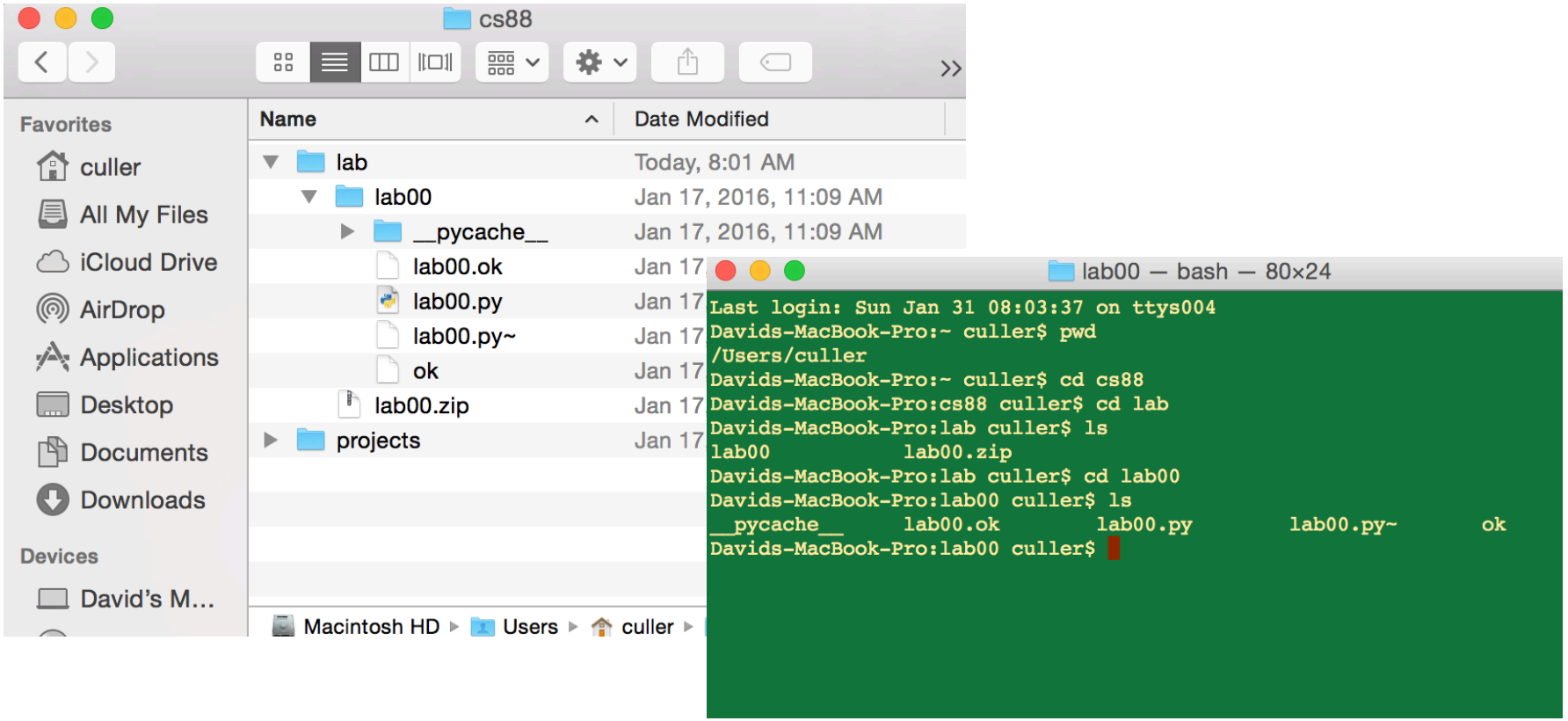
Lecture 2

January 25, 2016

Administrative issues

- Getting late enrollments into class
  - Your c8 account carries over
- HW1 due date deferred to Wed
- Labs are held in 273-5 Soda Mon 5-7
- Catch-up on Lab 0 today and start Lab 1
- HW2 is out
  - Defer due date to Tues?
- Concurrent enrollment students
  - Need email to get account set up and OK

Lab0: WIMP => Program Development



- **Big Idea: Layers of Abstraction**
  - **The GUI look and feel is built out of files, directories, system code, etc.**

Computational Concepts Toolbox

- Data type: values, literals, operations, e.g., int, float, string
- Expression `3.1 * 2.6`
- Call expression `max(0, x)`
- Variables
- Assignment Statement `x = <expression>`
- Sequences: tuple, list `(1,2), [3,4]`
  - numpy.array(<object>)
- Data structures
  - numpy.array, Table

- Tuple assignment `x,y = <exp>`

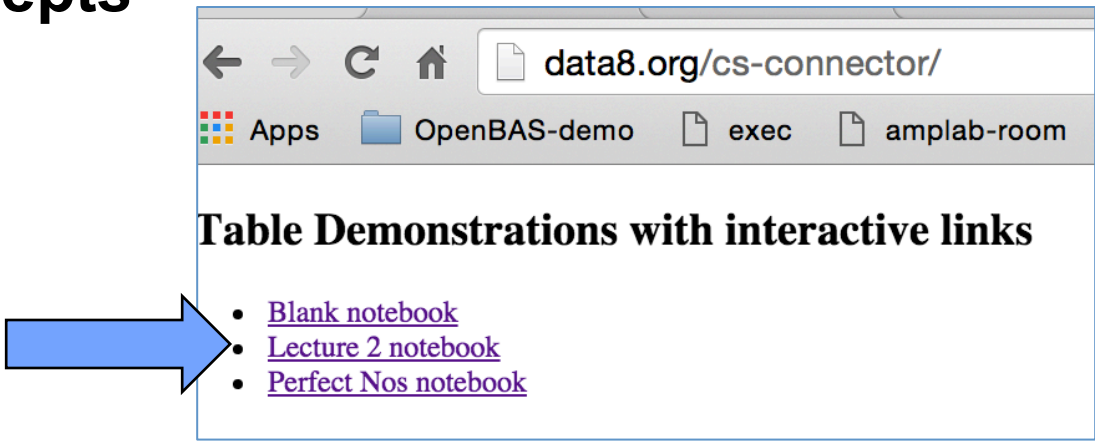
Computational Concepts today

- Call Expressions
- Function Definition Statement
- Conditional Statement
- Iteration: data-driven (list comprehension)
- Iteration: control-driven (for statement)
  - Structured
- Iteration: while statement
  - More primitive and more general

Big Idea: Software Design Patterns

"Philosophical" Context

- **Perfect Numbers**
  - A *perfect number* is a positive integer that is equal to the sum of its positive divisors, excluding itself.
  - e.g. 6 = 1 + 2 + 3
  - Euclid found the first 4 (the fifth found in the 1100s and 1400s
- **Proved N = $2^p - 1$ is prime (Mersenne Prime) then $(2^p - 1)2^{p-1}$ is even perfect**
- **Are there an infinite number of perfect numbers?**
- **Let's compute some while learning computational concepts**

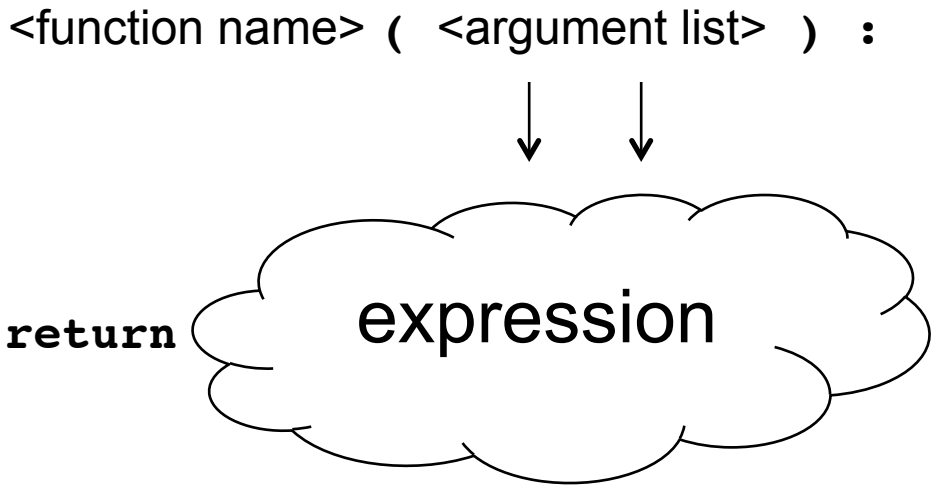


Call Expressions

- Evaluate a function on some arguments
- What would be some useful functions?

- builtin functions
  - https://docs.python.org/3/library/functions.html
  - min, max, sum
- https://docs.python.org/3/library/
- str
- import math; help(math)

Defining Functions



- **Generalizes an expression or set of statements to apply to lots of instances of the problem**
- **A function should *do one thing well***

Conditional statement

- **Do some statements, conditional on a *predicate* expression**

```
if <predicate>:
        <true statements>
else:
        <false statements>
```

Data-driven iteration

- **describe an expression to perform on each item in a sequence**
- **let the data dictate the control**

```
[ <expr with loop var> for <loop var> in <sequence expr > ]
```

`for` statement – iteration control

- **Repeat a block of statements for a structured sequence of variable bindings**

```
<initialization statements>
for <variables> in <sequence expression>:
  <body statements>

<rest of the program>
```

`while` statement – iteration control

- **Repeat a block of statements until a predicate expression is satisfied**

```
<initialization statements>
while <predicate expression>:
    <body statements>

<rest of the program>
```