

61A Lecture 1

Wednesday, August 23, 2017 (or perhaps even earlier)

Welcome to CS 61A!

You have two instructors this semester

John DeNero

denero@berkeley.edu

Office hours in 781 Soda
(starting next week)

Wed 2pm–3pm & Thurs 11am–12pm

By appointment: denero.org/meet.html

Paul Hilfinger

hilfinger@cs.berkeley.edu

Office hours in 787 Soda
(starting next week)

Time TBD

Best way to reach us: piazza.com/berkeley/fall2017/cs61a

Contact both of us & heads of staff: cs61a@berkeley.edu

The 61A Community

53 teaching assistants (TAs), formally known at Berkeley as UGSIs:

- Teach lab & discussion sections
- Hold office hours
- Lots of other stuff: develop assignments, grade exams, etc.

50+ tutors & mentors:

- Teach mentoring sections
- Hold office hours
- Lots of other stuff: homework parties, mastery sections, etc.

200+ lab assistants help answer individual questions & check your progress

1,900+ fellow students make CS 61A unique

Parts of the Course

Lecture: Videos posted to `cs61a.org` before each live lecture

Lab section: The most important part of this course (*next week*)

Discussion section: The most important part of this course (*this week*)

Staff office hours: The most important part of this course (*next week*)

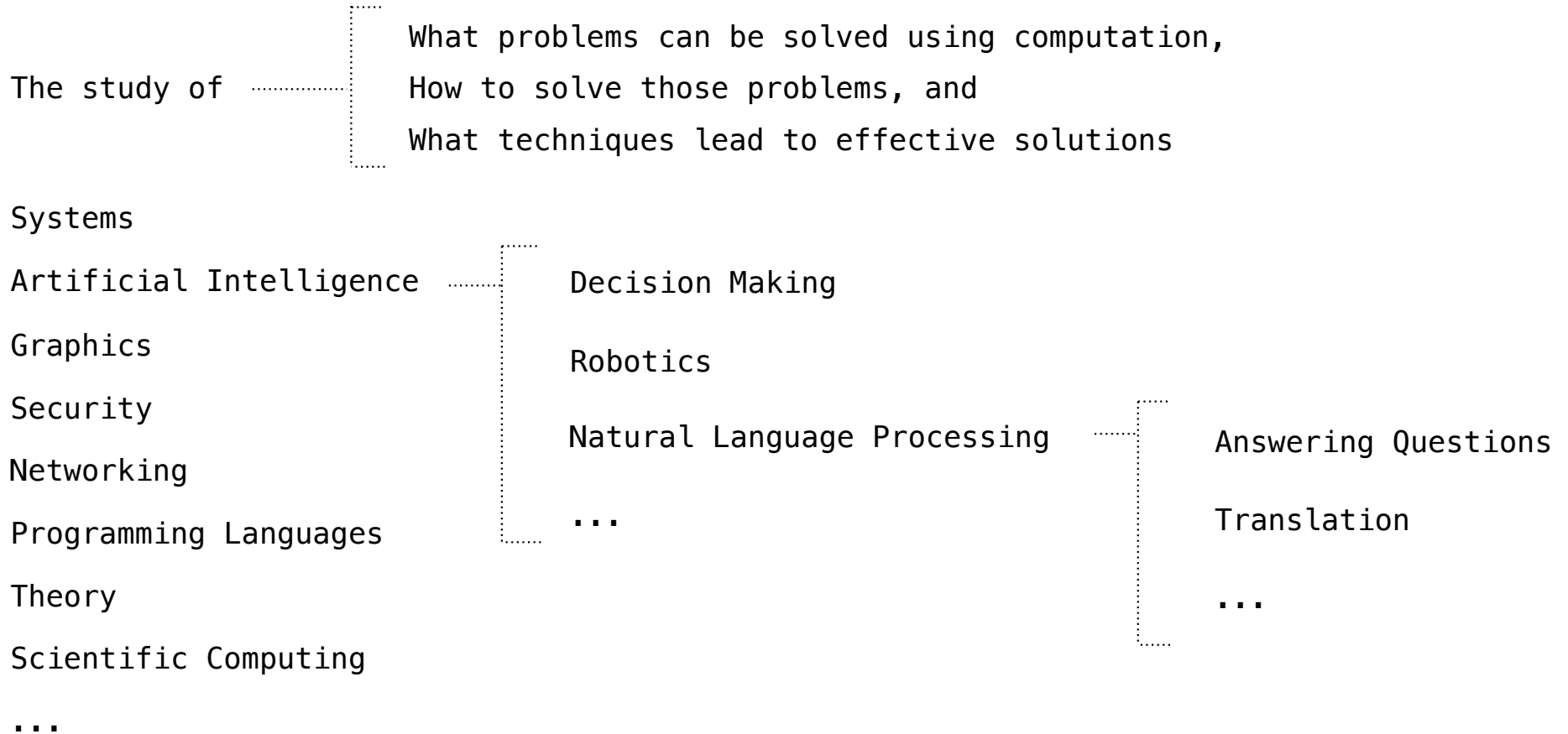
Online textbook: <http://composingprograms.com>

Weekly homework assignments, three exams, & four programming projects

Lots of optional special events to help you complete all this work

An Introduction to Computer Science

What is Computer Science?



What is This Course About?

A course about managing complexity

Mastering abstraction

Programming paradigms

An introduction to programming

Full understanding of Python fundamentals

Combining multiple ideas in large projects

How computers interpret programming languages

Different types of languages: Scheme & SQL

A challenging course that will demand a lot of you



Alternatives to CS 61A

CS 10: The Beauty and Joy of Computing

Designed for students without prior experience

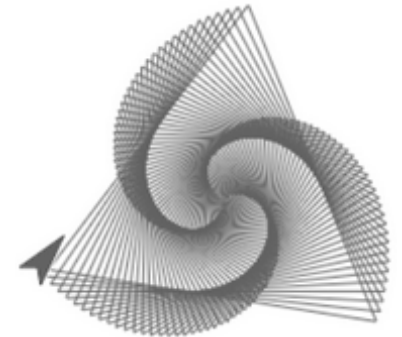
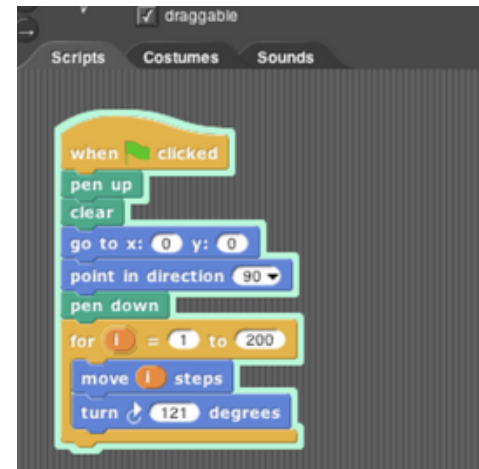
A programming environment created by Berkeley,
now used in courses around the world and online

An introduction to fundamentals (& Python)
that sets students up for success in CS 61A

Taught in Fall 2017 by Dan Garcia

50 seats available as of Tuesday 8/22
(but these will likely fill up)

More info: <http://cs10.org/fa17/>



Data Science 8: Foundations of Data Science

Fundamentals of computing, statistical inference, & machine learning applied to real-world data sets

More statistics than computer science

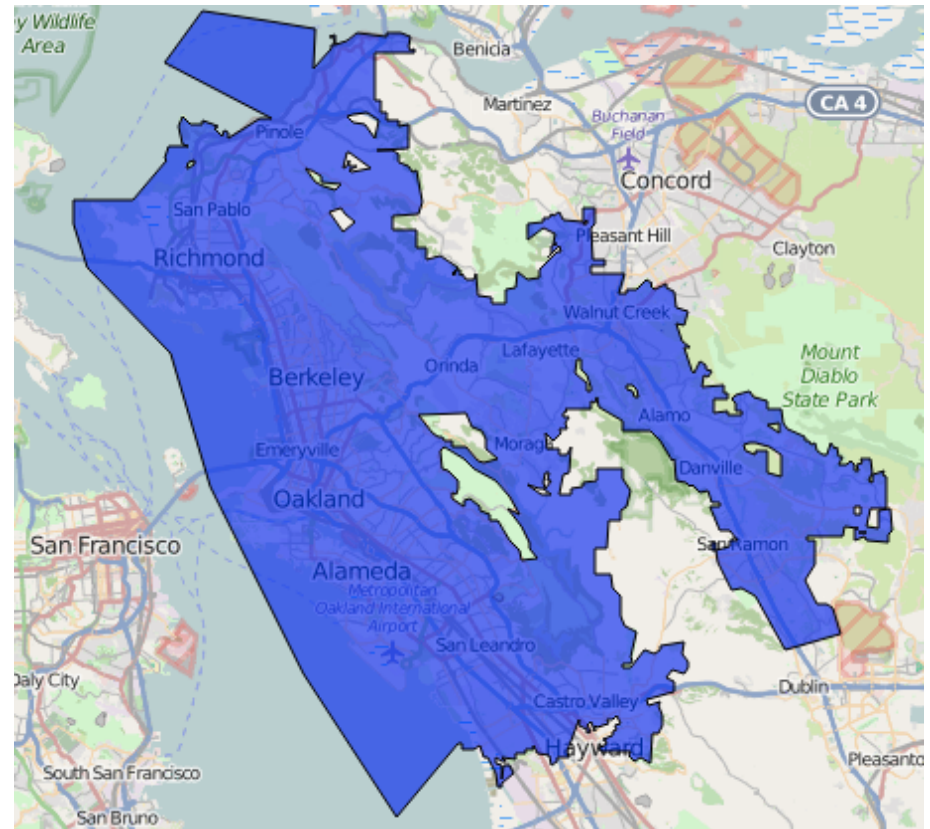
Great programming practice for CS 61A

Cross-listed as CS C8, Stat C8, & Info C8

Fall 2017: David Wagner & John DeNero

100+ seats available as of Tuesday 8/22

More info: <http://data8.org/fa17>



Course Policies

Learning
Community
Course Staff

Details...

<http://cs61a.org/articles/about.html>

Collaboration

Asking questions is highly encouraged

- Discuss everything with each other; learn from your fellow students!
- Projects 3 & 4 can be completed with a partner
- Choose a partner from your discussion section

The limits of collaboration

- One simple rule: Don't share your code, except with your project partner
- Copying project solutions causes people to fail the course
- We really do catch people who violate the rules, because...
 - We also know how to search the web for solutions
 - We use computers to check your work

Build good habits now

Expressions

Types of expressions

An expression describes a computation and evaluates to a value

$$18 + 69$$

$$\frac{6}{23}$$

$$\sin \pi$$

$$\log_2 1024$$

$$2^{100}$$

$$f(x)$$

$$\sqrt{3493161}$$

$$7 \bmod 2$$

$$\sum_{i=1}^{100} i$$

$$\lim_{x \rightarrow \infty} \frac{1}{x}$$

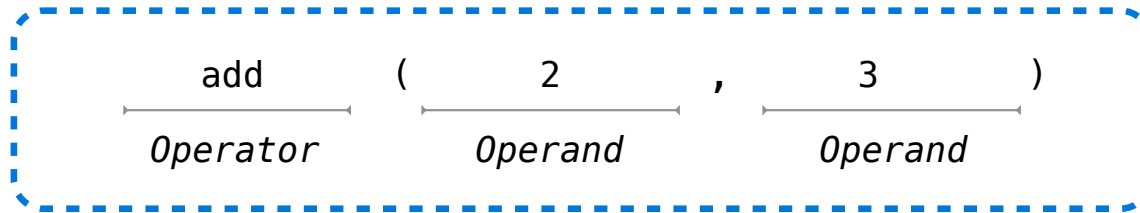
$$|-1869|$$

$$\binom{69}{18}$$

Call Expressions in Python

All expressions can use function call notation
(Demo)

Anatomy of a Call Expression



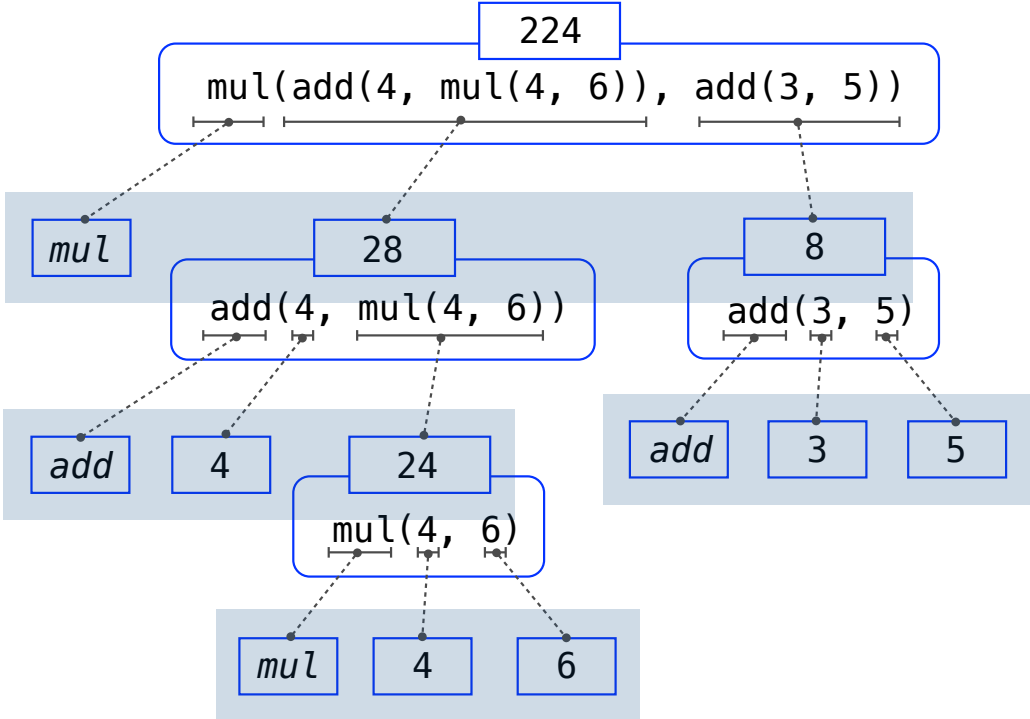
Operators and operands are also expressions

So they evaluate to values

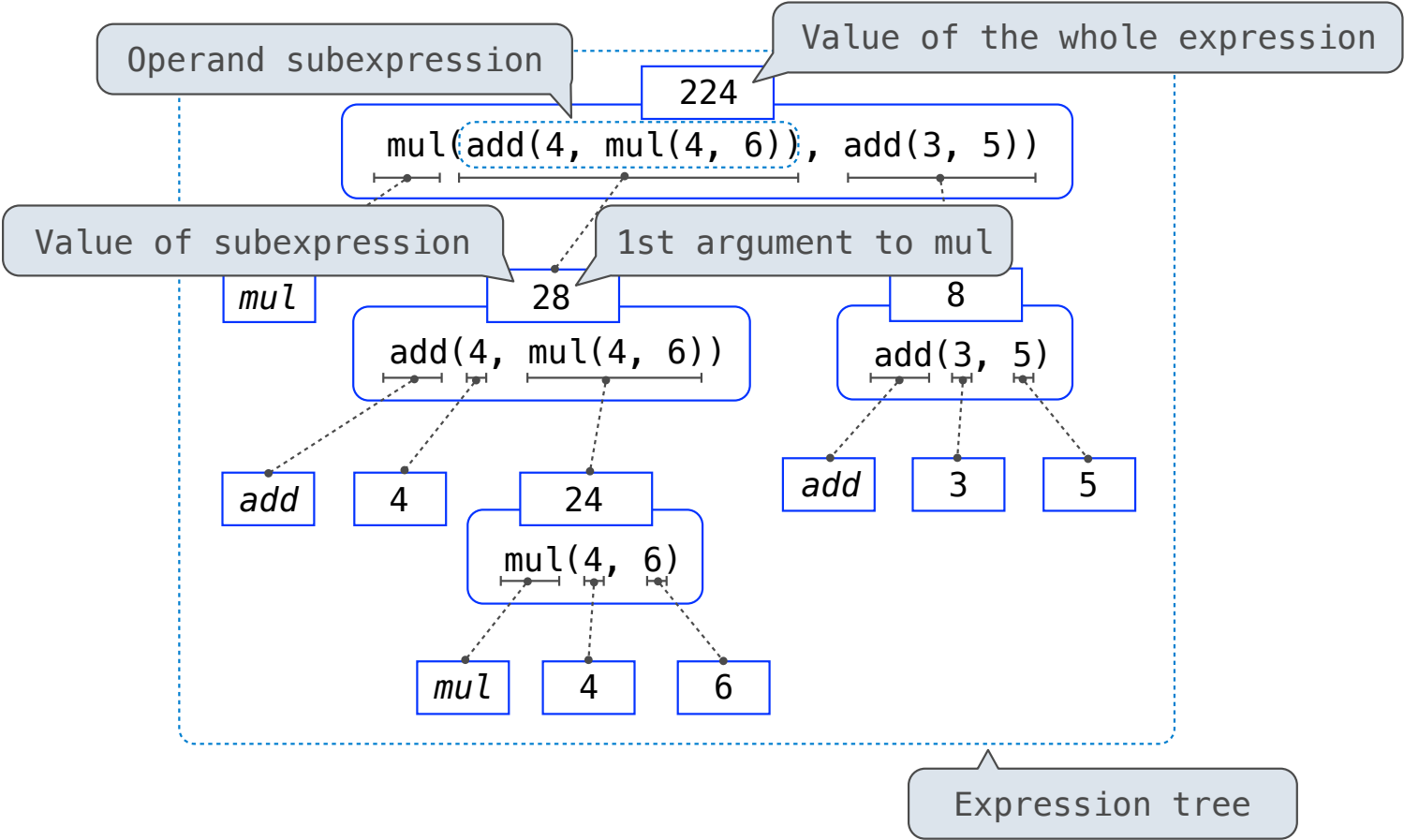
Evaluation procedure for call expressions:

1. Evaluate the operator and then the operand subexpressions
2. Apply the **function** that is the value of the operator subexpression to the **arguments** that are the values of the operand subexpression

Evaluating Nested Expressions



Evaluating Nested Expressions



Functions, Values, Objects, Interpreters, and Data

(Demo)