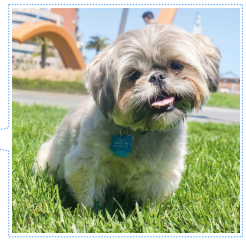
Functions

Welcome to CS 61A!

John DeNero
denero@berkeley.edu

Office hours in 781 Soda:
- 10:30–11:30 Wednesday (starting next week)
- 10:00–11:00 Thursday (starting this week)
- 11:00–12:00 Thursday online (see denero.org)
- 2:20–5:00 Monday 8/27 (next week only)
- By appointment: denero.org/meet.html



Fastest way to get answers: piazza.com/berkeley/fall2018/cs61a

Contact me, Alex, & Nancy: cs61a@berkeley.edu

The 61A Community

52 teaching assistants (TAs), formally known at Berkeley as GSIs or UGSIs:
- Teach lab & discussion sections
- Hold drop-in office hours
- Lots of other stuff: develop assignments, grade exams, etc.

50+ mentors:
- Teach mentoring sections
- Hold drop-in office hours
- Lots of other stuff: homework parties, mastery sections, etc.

250+ academic interns help answer individual questions & check your progress

2,000+ 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

Everything is posted to cs61a.org

An Introduction to Computer Science

What is Computer Science?

The study of
- What problems can be solved using computation,
- How to solve those problems, and
- What techniques lead to effective solutions

Systems
Artificial Intelligence
- Decision Making
- Robotics
- Natural Language Processing
  - Answering Questions
  - Translation
  - ...
- ...

Graphics
Security
Networking
Programming Languages
Theory
Scientific Computing
...

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

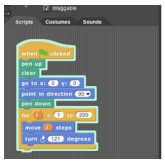
A programming environment created by Berkeley, now used in courses around the world and online

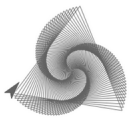
An introduction to fundamentals (& Python) that sets students up for success in CS 61A

Fall 2018: Dan Garcia

25 seats available

More info: http://cs10.org/fa18/





Data Science 8: Foundations of Data Science

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

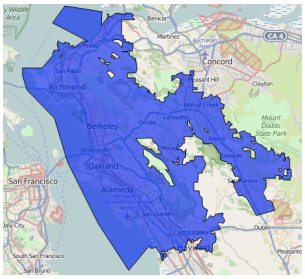
More statistics than computer science

Great programming practice for CS 61A

Listed as CS C8

180+ person waitlist

More info: http://data8.org/fa18



Course Policies

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!
- Some projects 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

$18 + 69$ $\quad \frac{6}{23}$ $\quad \sin \pi$ $\quad \log_2 1024$

2^{100} $\quad f(x)$ $\quad \sqrt{3493161}$

$7 \bmod 2$ $\quad \sum_{i=1}^{100} i$ $\quad \lim_{x \to \infty} \frac{1}{x}$

$|-1869|$ $\quad \binom{69}{18}$

Call Expressions in Python

All expressions can use function call notation

(Demo)

Anatomy of a Call Expression

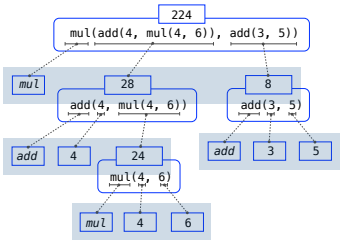
Operators and operands are also expressions

So they evaluate to values

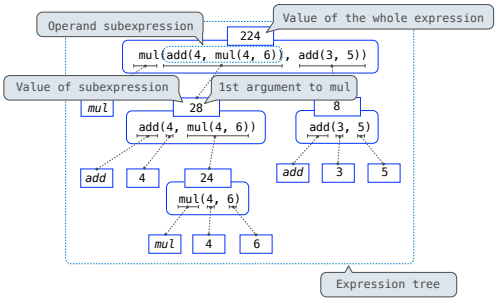
Evaluation procedure for call expressions:

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)