Welcome to CS 61A!

About Me

John DeNero
denero@berkeley.edu

20th semester teaching CS 61A

Teaching Professor in EECS & Faculty Director of Data Science Undergraduate Studies

Before that I was a Cal PhD student (2005–2010) and Google researcher (2010–2014)

Research focused on machine translation and how people interact with AI systems

My office hours start next week (Jan 27):

- Monday & Wednesday 3:15–4pm in 781 Soda Hall
- If that time doesn't work, email me and we'll set up an appointment

About the Course

What is This Course About?

A course about managing complexity

- Mastering abstraction
- Isolating and solving problems
- Techniques for organizing complex programs

An introduction to programming

- Full understanding of Python fundamentals
- Large projects to demonstrate how to manage complexity
- How computers interpret programming languages

Different types of languages: Python, Scheme, & SQL

cs61a.org

How to Succeed

Lecture, Videos, and the Textbook

Videos posted to cs61a.org are essential viewing **before** coming to lecture. All of the course content will be covered in the videos.

The textbook, composingprograms.com, is written to be concise and useful. Its content is very similar to the videos.

Lecture Mon, Wed, & Fri will review *the most important content* from the videos (but not all of it), work through examples, and discuss problem-solving strategies.

Student Advice from Fall 2024

"Watch videos before lecture"

"Watch the videos. Definetely helps with understanding the lecture beforehand, and the reason why I was so lost during the second half of the semester."

"Obviously watch the videos, try not to watch it at 2x speed (which is what I did and regret)...if you don't take time processing information, it will leave your brain by next morning"

"keep up with lectures, watch the videos, try to really understand everything along the way so it doesn't pile up at the end"

"Make sure to watch lecture videos before the lectures, so that lectures can be utilized for asking questions and further understanding."

https://cs61a.org/articles/advice/

Problem-Solving Practice

Solving problems becomes easier with **practice.**

Lab Monday/Tuesday: attendance is required (unless you're in mega lab)

Discussion on Wed/Thurs/Fri: attendance is required (unless you're in mega discussion)

These prepare you for weekly **homework** assignments & 4 larger programming **projects**

Drop-in one-on-one assignment help (called "**office hours**" at Cal) starts next week.

What does a "discussion section" look like?

Expectation

https://engineering.berkeley.edu/students/academic-support/

Reality

https://www.microsoft.com/en-us/research/blog/grassroots-data-science-education-uc-berkeley/

Goal: Provide a great environment to learn how to solve problems through practice & *discussion*

I've seen small groups of Cal students do amazing things!

Discussion (Starts This Week)

Unless you've elected the mega discussion...

- You should have a group number shared with the 4–6 students in your group, a room, and a discussion time. There will be 2–7 groups per room, so make sure you find the right group.
- TAs often run discussions across multiple rooms simultaneously.

What happens during discussion section?

- You're given a worksheet full of example problems to solve together & some instructions.
- The point is not just to solve those problems, but to learn how to solve similar problems.
  - Discussion problems aren't graded; you don't have to solve them all.
- Bring a laptop or tablet, but a phone works in a pinch.

Lab 0: Getting Started

Lab 0 is posted and should be completed **before** you come to Lab 1 next Monday/Tuesday.

Drop-in office hours to get help with Lab 0 are on Friday 10am–12pm in Soda 310.

https://cs61a.org/office-hours/

Asking Questions

?

Ed: You can reach all staff (private posts) and all students (public posts)

denero@berkeley.edu: Don't be surprised if I ask you to post on Ed

cs61a@berkeley.edu: Goes to several staff members

Student Advice from Fall 2024

"Try to collaborate and with others and try to make friends within the class."

"Attend lab and make sure you understand how each program is structured"

"really utilize discussions. learning is easier with others!"

"Attend discussions and labs as it helps you discuss the content with other students"

"Go to discussion. I am so, so grateful for the fact that I had an active discussion group. Make the tough choice and commit yourself to spending that hour each week solving problems with other people just as confused as you are-- I guarantee that you'll look back on the decision and have no doubt that it was worthwhile."

"ASK OTHER PEOPLE FOR HELP WHEN NEEDED, DON'T BE AFRAID TO MAKE FRIENDS!!!"

Should you take CS 61A?

According to the Syllabus: cs61a.org/articles/about/

There is no formal programming-related prerequisite for CS 61A, but...

- Taking the course without any prior programming experience is typically quite challenging.
- Most CS 61A students have had significant prior programming experience.
- Students who take the course without prior programming experience typically must spend more time to complete assignments and tend to receive lower final grades in the course.

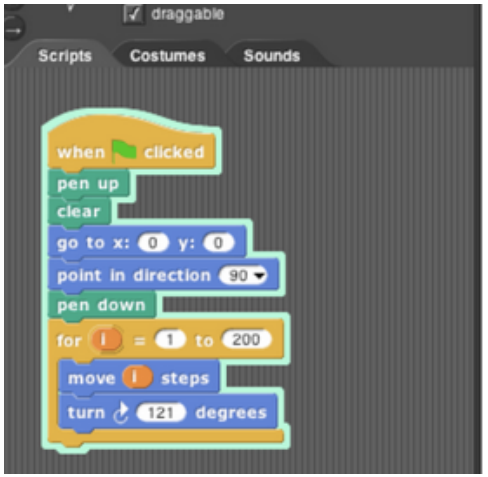
CS 10: The Beauty and Joy of Computing

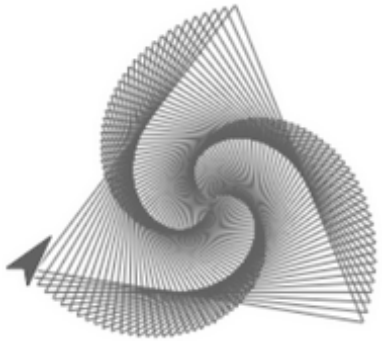
Designed for students without prior experience

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 and Data C88C

More info: http://cs10.org/





Data C88C (aka CS 88): Computational Structures in Data Science

Based on CS 61A, but covers only 3 out of 4 units worth of the content:

- Two programming projects (instead of four)
- Everything you need to know to continue on to CS 61B
- Omits the unit on how programs run other programs & a few advanced Python topics

Designed for students taking Data 8 (Foundations of Data Science), but is now independent

The course is full, but the waitlist is pretty short at the moment.

Student Advice from Fall 2024

"If you have no coding experience this is really hard"

"Do not take this class if you don't have programming experience!"

"Make sure you have previous programming experience or understand the contents in AP CSA well before taking this class. It will be very challenging and time consuming if you are new to programming."

"As someone who came into this class with virtually no formal coding experience, remember that it doesn't mean you can not do well in the class. You just need to be prepared to put much more work in than others who do have experience."

"Need prior coding experience to succussed in this class."

"If you have no experience, it's doable, undeniably hard, but doable."

Course Policies

Course Policies

Learning

Community

Course Staff

Details...

https://cs61a.org/articles/about/

Collaboration

Working together is highly encouraged

- Discuss everything with each other; learn from your fellow students!
- Some projects can be completed with a partner

What constitutes academic misconduct?

- *Please* don't look at someone else's code!
  Exceptions: lab, your project partner, or **after you already solved the problem.**
- *Please* don't tell other people the answers! You can point them to what is wrong and describe how to fix it or show them a related example.
- *Please* don't use ChatGPT or other similar tools to write code for you.
- Copying project solutions causes people to fail the course.

Build good habits now

Student Advice from Fall 2024

"LLMs are useful tools that you'll probably use on occasion as a software engineer or computer scientist, but you should be able to solve problems on your own without ChatGPT, as someone who doesn't need to rely on ChatGPT but can use it to augment their workflow will be significantly more productive and competent than someone who is reliant on ChatGPT."

"It'll be very tempting at points to just ask ChatGPT when you're running into a bug. For your sake, I strongly recommend against it... The skills you gain from solving the more simple problems in CS61A will be essential in your ability to solve tougher problems later on, and so reliance on ChatGPT can only take you so far. If you do ever use ChatGPT because you're rushing to meet a deadline, I strongly recommend coming back to the question later and thoroughly understanding it."

Let's Stop Harassment & Discrimination

Disparaging remarks targeting a particular gender, race, or ethnicity are not acceptable.

From the Berkeley Principles of Community:

> "We affirm the dignity of all individuals and strive to uphold a just community in which discrimination and hate are not tolerated."

From the EECS department mission:

> "Diversity, equity, and inclusion are core values in the Department of Electrical Engineering and Computer Sciences. Our excellence can only be fully realized by faculty, students, and staff who share our commitment to these values."

denero.org/feedback.html: If you want to stay anonymous but make me aware of something happening in the course.

EECS Student Climate & Incident Reporting Form: Informs the EECS department of any issues. You can also contact Susanne Kauer (skauer@berkeley.edu) directly.

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}

$f(x)$

$\sqrt{3493161}$

$7 \bmod 2$

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

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

$|-1869|$

$\binom{69}{18}$

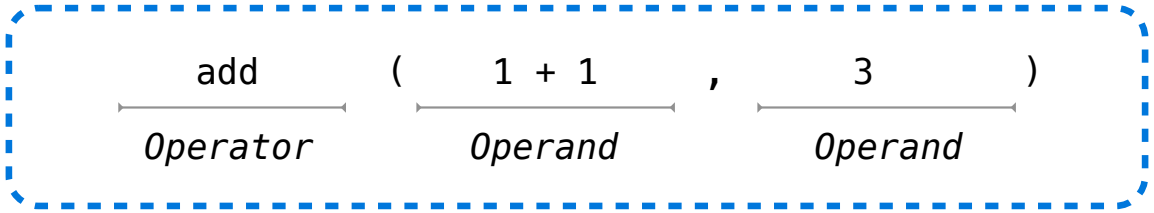
Call Expressions in Python

All expressions can use function call notation

(Demo)

Call Expressions

Anatomy of a Call Expression



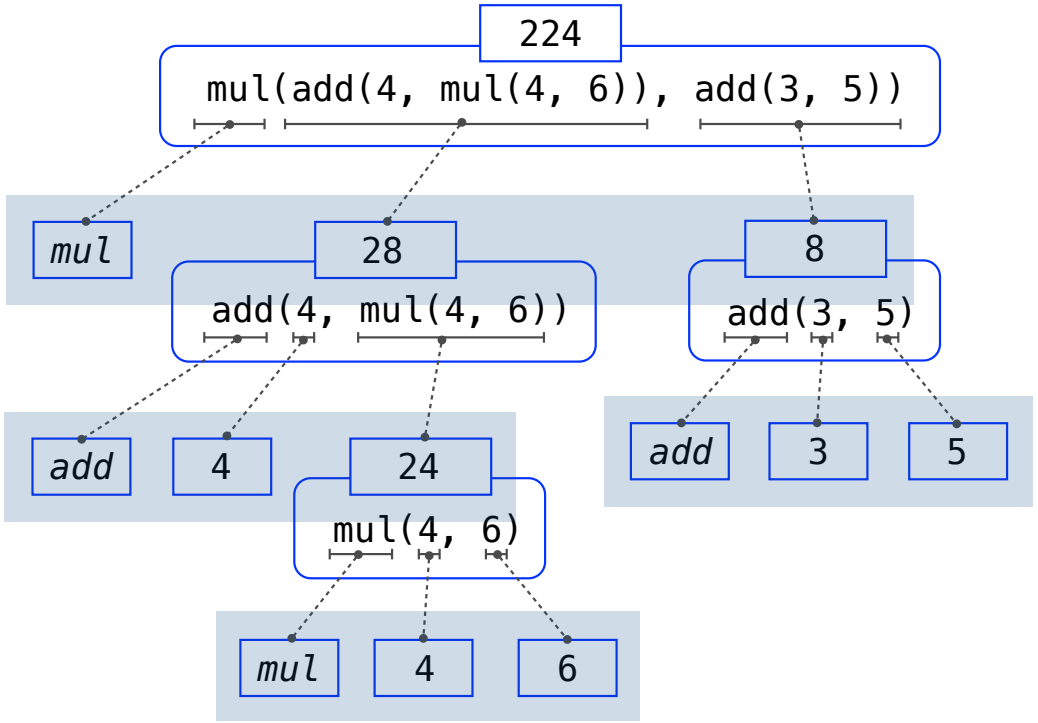
Operators and operands are also expressions

Expressions evaluate to values

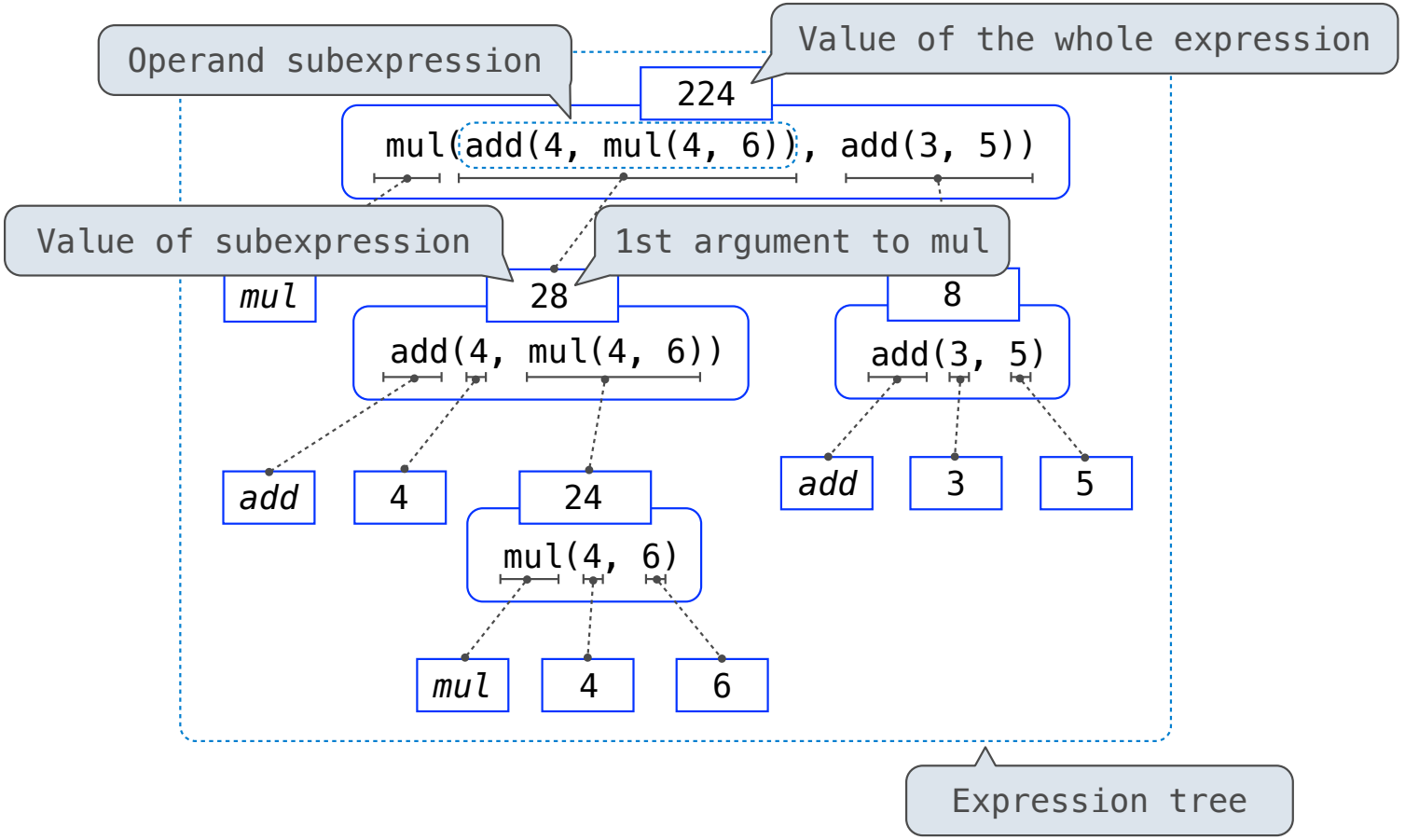
Evaluation procedure for call expressions:

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

Evaluating Nested Expressions



Evaluating Nested Expressions



Demo

```
python3 ok --local -q python-basics -u
```