

# Welcome to Data C88C!

---

Monday, June 23rd, 2025

Week 1

Summer 2025

Instructor: Eric Kim ([ekim555@berkeley.edu](mailto:ekim555@berkeley.edu))

# Lecture 1 Overview

---

- Course overview
- Online course setup
- Teaching staff introduction
- Course policies, administtrivia

# Announcements

---

- Welcome to Lecture 1!
  - Lectures and labs start during Week 1
    - **First Lab**: Tuesday June 24th
  - Office hours **do not start until Week 2** (June 30th).
  - Important Ed posts
    - [Welcome Ed post](#)
    - [Course index](#) (will be updated throughout the course)
    - Please, please read these!
-

## (tldr reference) Important Links

---

- **Course website:** <https://c88c.org/su25/>
  - All assignments are released here.
- **Gradescope:** <https://www.gradescope.com/courses/1053326>
  - You submit assignments here, and receive grades here too
- **Ed:** <https://edstem.org/us/courses/79702/discussion>
  - Important course announcements are posted here.
  - You can ask questions here too
- **bCourses:** <https://bcourses.berkeley.edu/courses/1545181>
  - All course Zoom links are here: [\[link\]](#)
  - Recorded lecture/lab videos are here [\[link\]](#)
- **Tip:** for the above links, use your @berkeley.edu email address (“CalNet ID”) to log in (or via “CalNet” SSO)

**Pro Tip:** bookmark these links!

---

# Online course setup

---

- Data C88C su25 is taught “**fully online**”
  - What does this mean for you (the student)?
    - Lectures, labs, office hours, and exams done through **Zoom**
    - **You do not need to physically be present in Berkeley, CA for this course**
  - **Attendance**: we don't track attendance for lecture or lab. Attendance is not part of your grade.
    - That said: student success correlates strongly with active participation in class activities.
-

# Online course setup

---

- **Where to find the Zoom links?**
    - [bCourses Zoom page](#) has ALL course Zoom links
      - Also here: "Course Zoom links" Ed post: [\[link\]](#)
    - Lectures, Labs, OH
    - I'll put them in an Ed post too
  - **Where to find/submit assignments?**
    - [Course website](#) will have all assignments
    - Submit ALL assignments (lecture quizzes, labs, hws, projects) to [Gradescope](#)
      - You will get your assignment scores here too!
    - Tip: Gradescope has assignment due dates as well!
  - **Where to ask questions?**
    - [Ed](#) is an online course forum. Use this liberally!
    - Note: feel free to answer questions as well!
      - "The best way to learn something is to be able to teach it"
-

# Online course setup: Ed

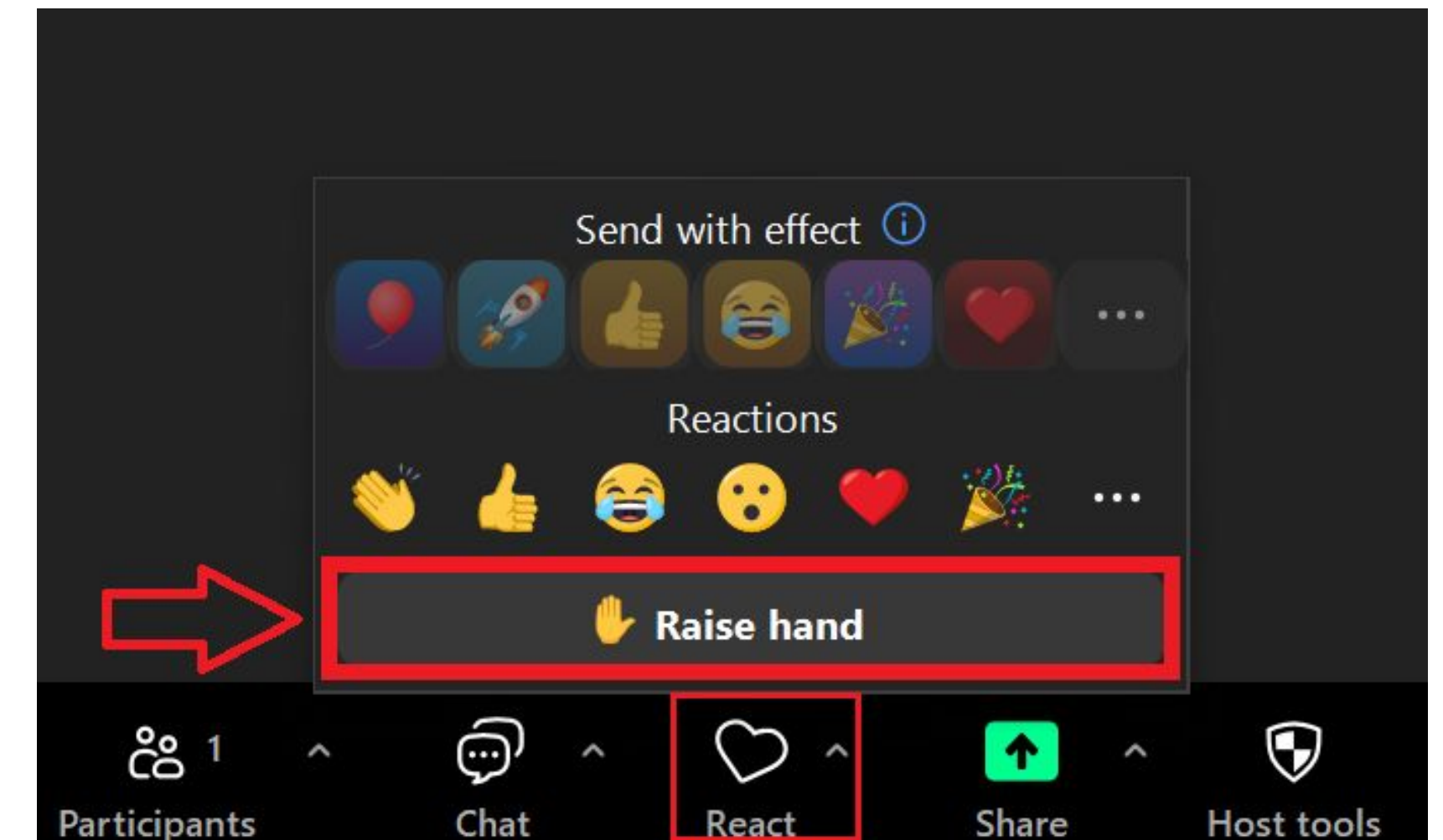
---

- Important: our **primary means of communication to you** (the students) will be through **Ed**
- It's **your responsibility** to follow the above pages to keep up to date with things like: assignment releases/deadlines, exam details, etc.
  - I will be very sad if you miss an announcement and, say, miss a project deadline



# Online course setup: Asking questions during lecture (Zoom)

- **Option 1:** "Raise hand" button
- **Option 2:** ask in Zoom chat
  - Note: Zoom text chats are saved after each lecture (along with author names). So, please keep the text chat respectful and professional (eg don't write anything you wouldn't want your parents to see)



<https://tactiq.io/learn/how-to-raise-your-hand-in-zoom-meetings>



## Online course: Exams

---

- **Midterm:** Tuesday July 15th 2025 from 3 PM - 5 PM PST.
- **Final exam:** Tuesday August 12th 2025 from 3 PM - 5 PM PST.
- Both exams will be **done online (Gradescope)**, with **Zoom proctoring**.
  - Details will be released closer to the exam date.
- For those that can't make these exam times: we will send another announcement closer to the exam dates to handle conflicts.

# “Berkeley Time”

---

- **“Berkeley time”**: lectures/labs start **10 minutes AFTER** the posted start time.
  - Example: Lecture posted start time is 3pm, but I’ll actually **start the lecture at 3:10pm**.
- I’ll still start the Zoom lecture meeting at 3pm. Feel free to join early and treat it as an informal office hours, eg asking questions, chatting about random things, etc.
- Important: **exam times** will start promptly at the posted start time.

# About Me

---

**Eric Kim:** [ekim555@berkeley.edu](mailto:ekim555@berkeley.edu) (more about me [here](#))

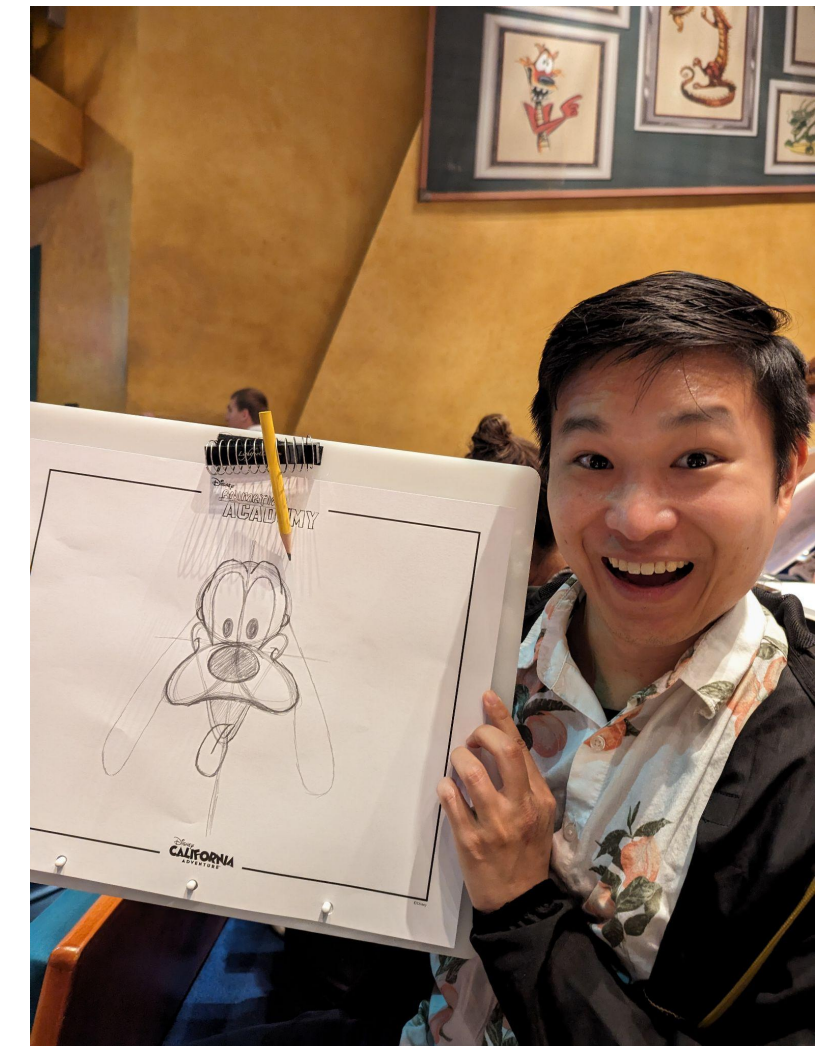
## **Brief history:**

UC Berkeley (2011, L&S CS), UCLA (2016, MS with focus in AI/ML and computer vision).  
ML engineer at Pinterest Visual Search team (2017 to now!).

**What I do at work:** ML modeling (object detection, representation learning, recommendation systems), ML serving infra (training/serving, GPU), big-data pipelines, and more...

**What I teach:** Data C88C, Data C182 (Intro to Deep Learning)

**What I do for fun:** Music (piano, guitar, bass, vocals. Rock/pop/blues/jazz/classical), video games, running (casually). I love [Pixar](#) movies and animation in general!





# Meet the staff!

---

## TA's



**Mia Lopez** she/her/hers

@ [mglopez@berkeley.edu](mailto:mglopez@berkeley.edu)

👋 hello! My name's Mia and I'm a 4th year studying CS! I'm into more indie music, like Childish Gambino or TV Girl. My favorite movies are Arrival and Lego Batman, but I also have a thing for Wes Anderson movies! This is my 3rd semester on CS61A staff, so feel free to reach out whenever! 🐾



**Aneesh Durai** he/him/his

[aneesh.durai@berkeley.edu](mailto:aneesh.durai@berkeley.edu)



**Mira Wagner** she/her/hers

[mirawagner@berkeley.edu](mailto:mirawagner@berkeley.edu)

Hi! I am a junior majoring in data science and linguistics. I love reading, especially mysteries, swimming and baking! Excited for this semester :)



**Satleen Gill** she/her/hers

[satleen@berkeley.edu](mailto:satleen@berkeley.edu)

Hello! I am a rising senior double majoring in Data Science and Computer Science. I love to read, paint, go to concerts, and travel! Excited for the summer session!

## Tutors



**Haoming Chen** he/him/his

[haoming\\_chen@berkeley.edu](mailto:haoming_chen@berkeley.edu)

Hi everyone, I am Haoming, a rising sophomore interested in soccer, data science, hiking, science fiction, machine learning in Physics, and reach out to me to talk about any of these or anything in between!



**Philip Ngo** he/him/his

[philippngo@berkeley.edu](mailto:philippngo@berkeley.edu)

Hi everyone! I'm Philip, a rising 3rd year majoring in Data Science and Linguistics from Long Beach, CA. I love summers in SoCal; some of my favorite summer activities are the going to the beach, karaoke, and concerts (saw gretperez and Malcom Todd this summer, seeing Laufey this fall!). Looking forward to seeing you all this summer!

Learn more about your TA's + Tutors here: <https://c88c.org/su25/staff/>

---

# 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

Different types of languages: Python, SQL

How to Succeed



# Lecture, Videos, and the Textbook

Videos posted to <https://c88c.org/su25/> are essential viewing **before** coming to lecture. All of the course content will be covered in the videos.

The textbook, [composingprograms.com](https://composingprograms.com), is written to be concise and useful. Its content is very similar to the videos.

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

**Important:** watch these videos before lecture to maximize learning!



## Calendar

Week	Date	Lecture	Textbook	Lab & Discussion Links	Homework & Project
1	Mon 6/23	Welcome		Disc 00: Getting Started	
	Tue 6/24	Functions <a href="#">Videos</a>	<a href="#">Ch. 1.1</a> <a href="#">Ch. 1.2</a> <a href="#">Ch. 1.3</a>	Disc 01: Functions  Lab 00: Getting Started <b>Due Sun 6/29</b>	HW 01: Functions <b>Due Sun 6/29</b>
	Wed 6/25	Control <a href="#">Videos</a>	<a href="#">Ch. 1.4</a> <a href="#">Ch. 1.5</a>	Disc 02: Control, Environment Diagrams	



# Student Advice from Fall 2024

---

"Watch videos before lecture"

"Watch the videos. Definitely 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://c88c.org/su25/articles/advice/>

# Problem-Solving Practice

---

Solving problems becomes easier with **practice**.

**Lab** Tuesday/Thursday: exercises. **Graded**.

**Discussion notes**: additional practice + notes. **Ungraded, optional**.

These prepare you for **homework** assignments & the larger programming **projects**

Drop-in one-on-one assignment help (called "**office hours**" at Cal) starts during week 2 (June 30th 2025).

**Office hours offered by**: TA's, Tutors, Instructors

# Grading policy

---

This course is not curved, and has point bins. For details, see the course website "Syllabus": [\[link\]](#)

(2025-06-23) I may add a small "pre midterm" assignment, will update on Ed + update the Grading policy if this happens

## Grading

Your course grade is computed using a point system with a total of 200 points, broken down as follows:

- The midterm, worth 50 points
- The final exam, worth 75 points
- Two projects, worth 45 points
- Homework, worth 20 points
- Lab, worth 10 points

There are 2 extra credit points available to everyone for early submission of projects.

Each letter grade for the course corresponds to a range of scores:

	A	≥ 185	A-	≥ 170	
B+	≥ 155	B	≥ 140	B-	≥ 125
C+	≥ 115	C	≥ 105	C-	≥ 95
D+	≥ 90	D	≥ 85	D-	≥ 80

## 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"

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

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

Should you take Data C88C?

## According to the Syllabus: <https://c88c.org/su25/articles/about-c88c/>

---

There is no formal programming-related prerequisite for Data C88C, but...

- Taking the course without any prior programming experience is typically quite challenging.
- Most Data C88C 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.
- We recommend you have taken **DATA 8** or are concurrently taking DATA 8. This definitely helps with the programming practice.



# CS 10: The Beauty and Joy of Computing

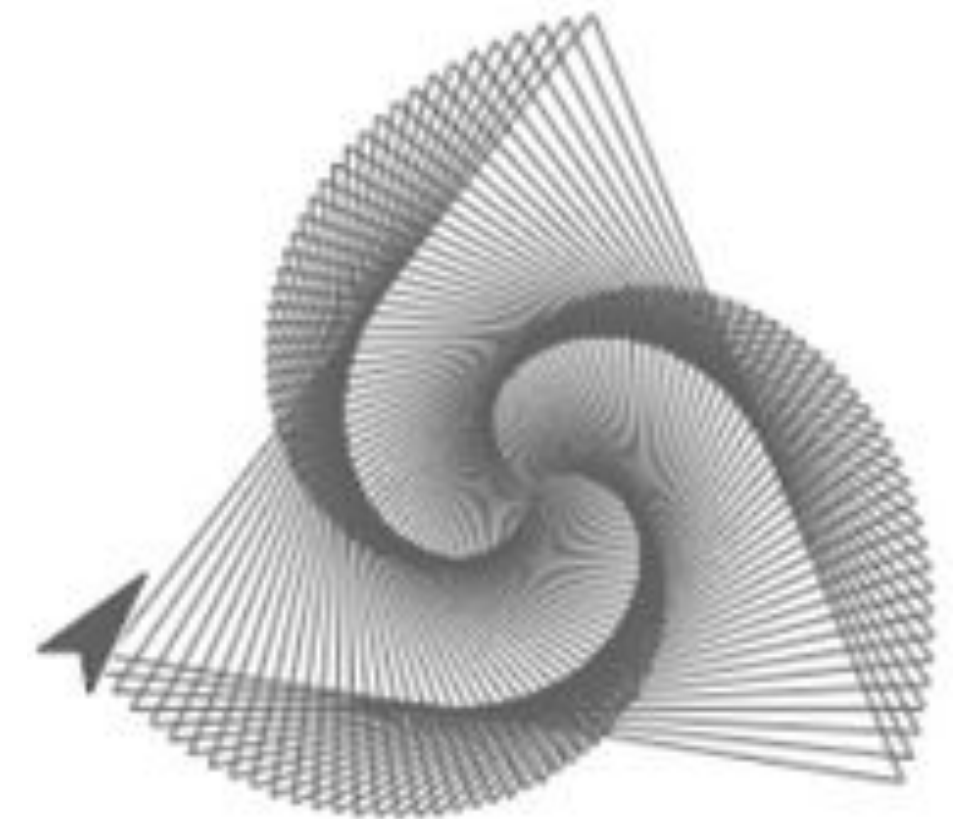
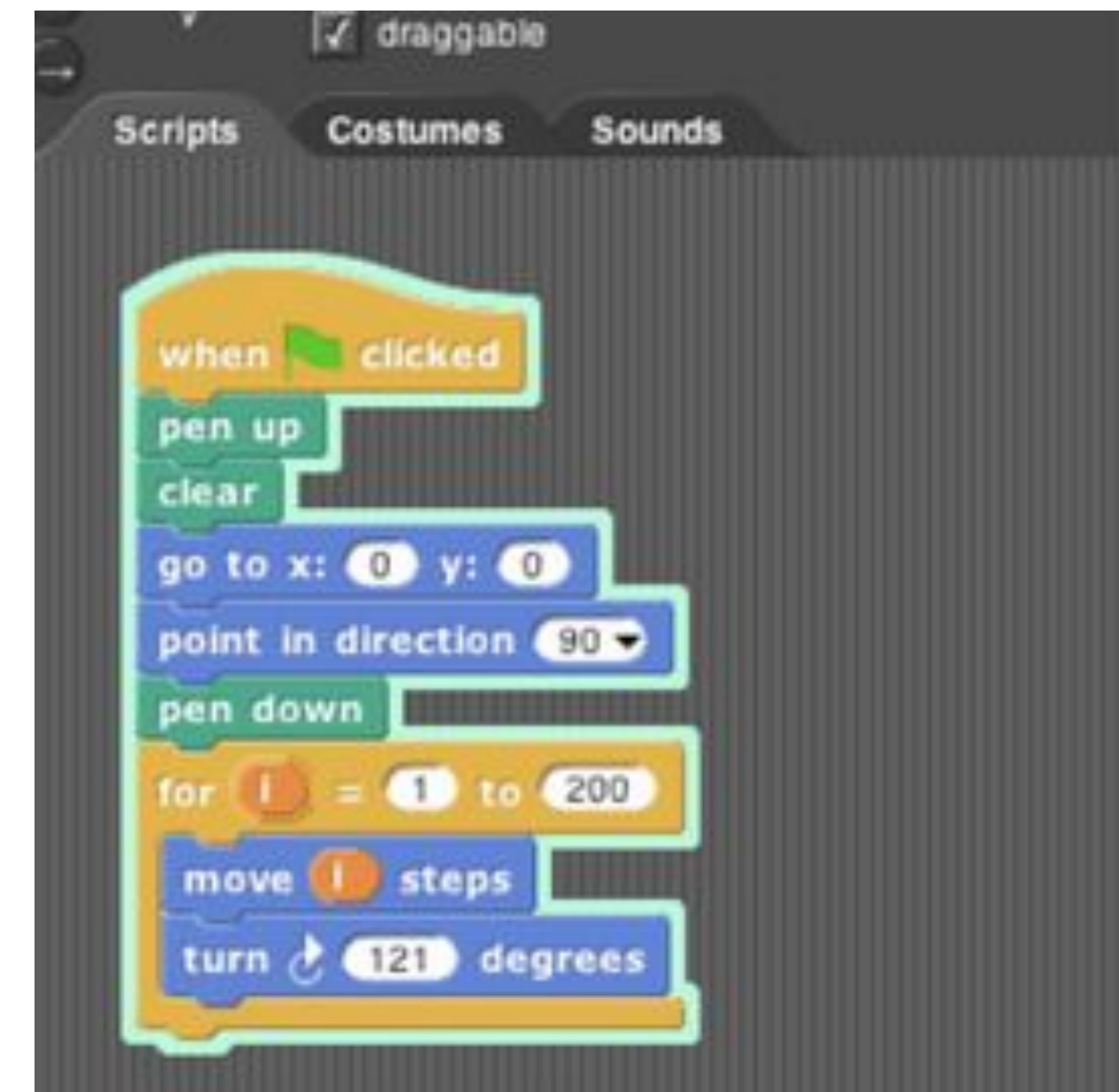
---

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 vs CS 61A

---

Data C88C is 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 (interpreters) & a few advanced Python topics



# Course Policies

Learning

Community

Course Staff

Details...

<https://c88c.org/su25/articles/about-c88c/>

# 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."

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

# Quick Python Intro: terminal, interpreters, and expressions

# What is a terminal?

---

In short: a terminal is a **text-based way of interacting** with your computer.

Windows: PowerShell

MacOS: iTerm/iTerm2

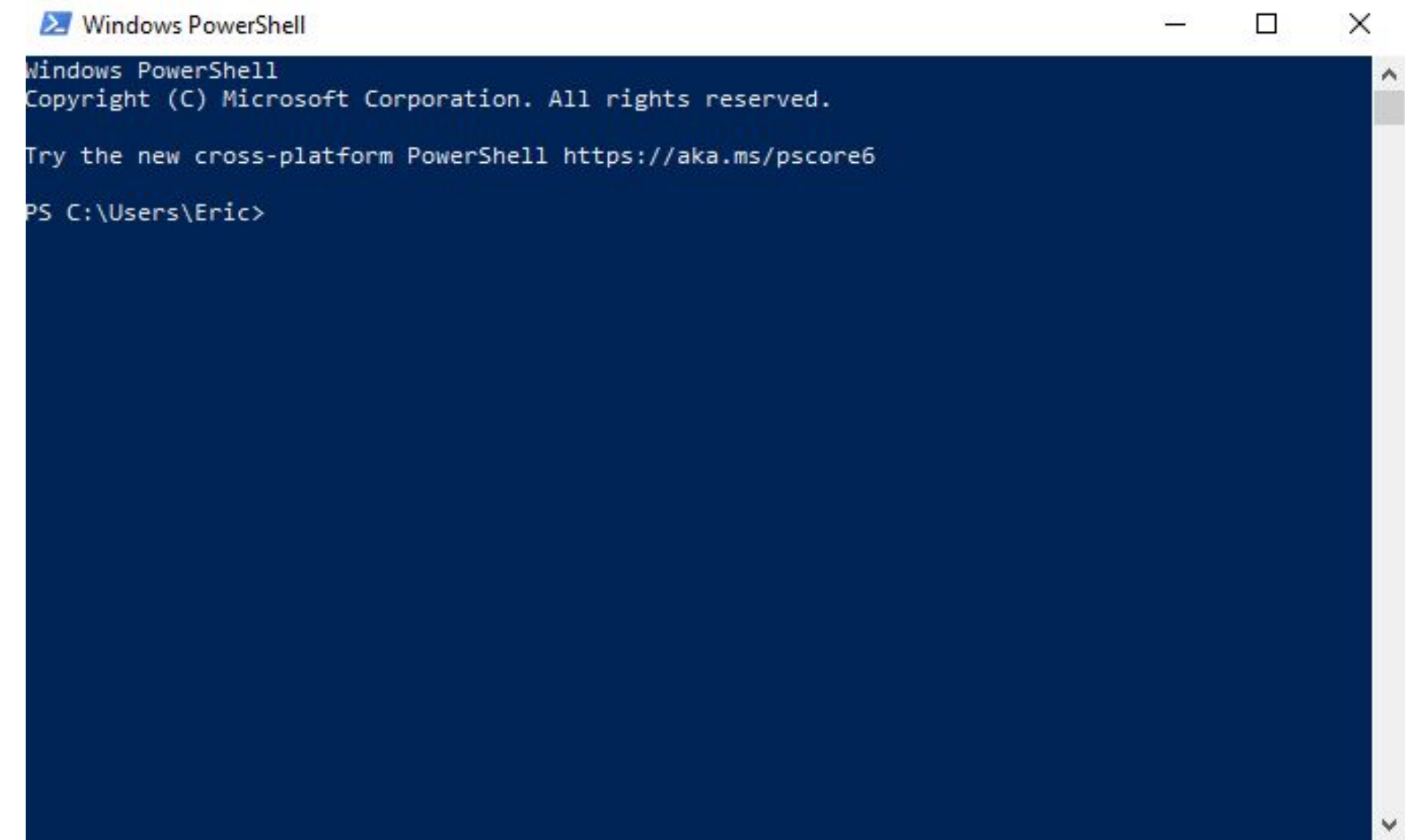
Linux: Bash/zsh

**Important concepts:**

**File systems.** Files, folders, how to move to different directories.

**Commands.** Ex: the "ls" command shows all files/folders in the current directory.

Note: you will cover this in Lab00

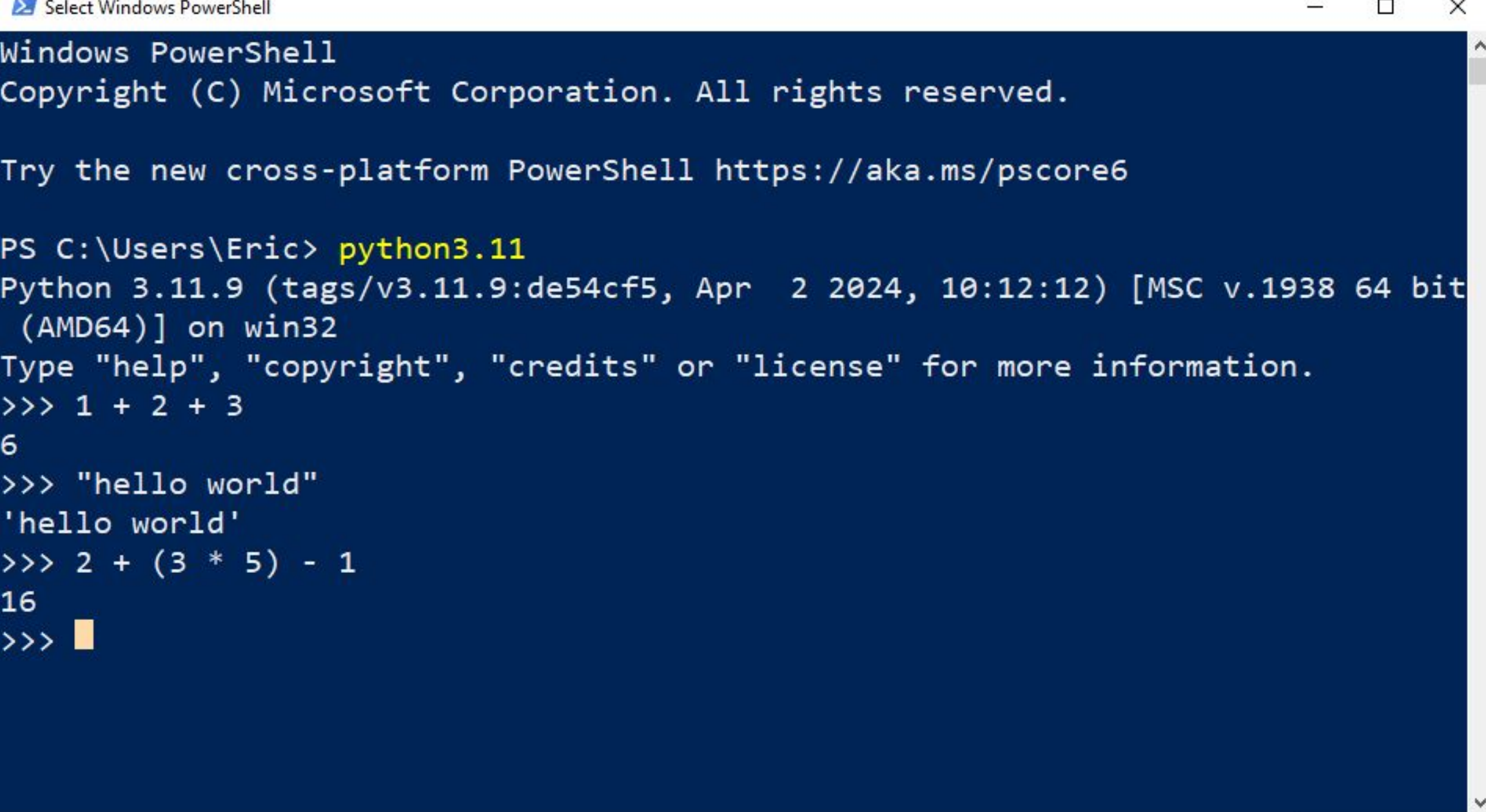




# What is an interpreter?

---

- A Python interpreter is an interactive, text-based way of running Python programs.
- To enter the **Python interpreter** from your **terminal**, you'll typically run a command like: `python`, `python3`, or `python3.11`
- In the Python interpreter, you can **type in a Python expression**, and the interpreter will **evaluate** the expression, and **show (print)** you the **expression's value**.



```
Select Windows PowerShell

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\Eric> python3.11
Python 3.11.9 (tags/v3.11.9:de54cf5, Apr  2 2024, 10:12:12) [MSC v.1938 64 bit
(AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> 1 + 2 + 3
6
>>> "hello world"
'hello world'
>>> 2 + (3 * 5) - 1
16
>>> █
```



# Types of expressions

---

An **expression** describes a computation and **evaluates to a value**

$$18 + 69$$

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

$$\sin \pi$$

$$\log_2 1024$$

$$2^{100}$$

$$f(x)$$

$$7 \bmod 2$$

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

$$\sqrt{3493161}$$

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

$$|-1869|$$

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

## Expressions (and their Python equivalents)

---

$18 + 69$     `>>> 18 + 69`  
87

$2^{100}$     `>>> 2 ** 100`  
1267650600228229401496703205376

$\sin \pi$     `>>> from math import pi`  
          `>>> from math import sin`  
          `>>> sin(pi)`  
1.2246467991473532e-16

Note: 1.22e-16 means:  $1.22 * 10^{-16}$  (aka a number very close to 0).  
(For fun) why isn't `sin(pi)` exactly 0? If you're curious: [\[link\]](#). tldr: computers can't represent numbers like pi with exact precision.

---

# Call Expressions in Python

---

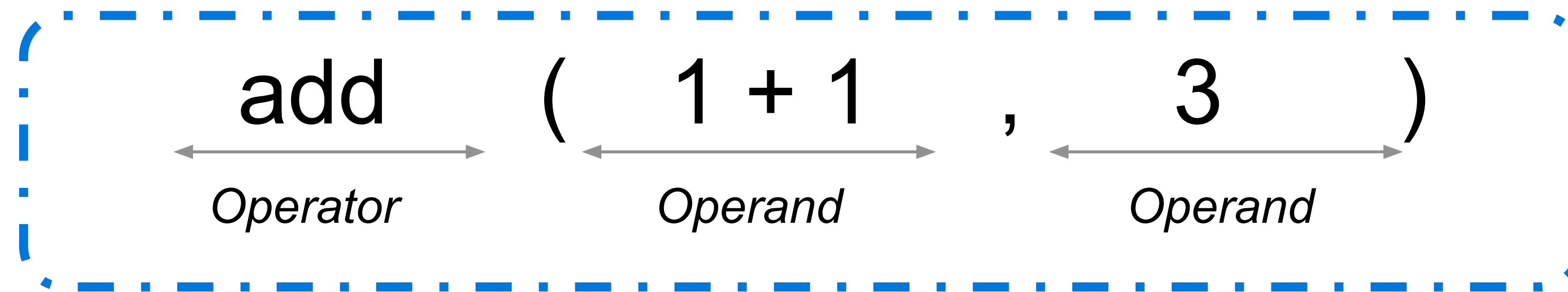
All expressions can use function call notation

(Demo: 01.py:Demo00)

# Call Expressions

# Anatomy of a Call Expression

---



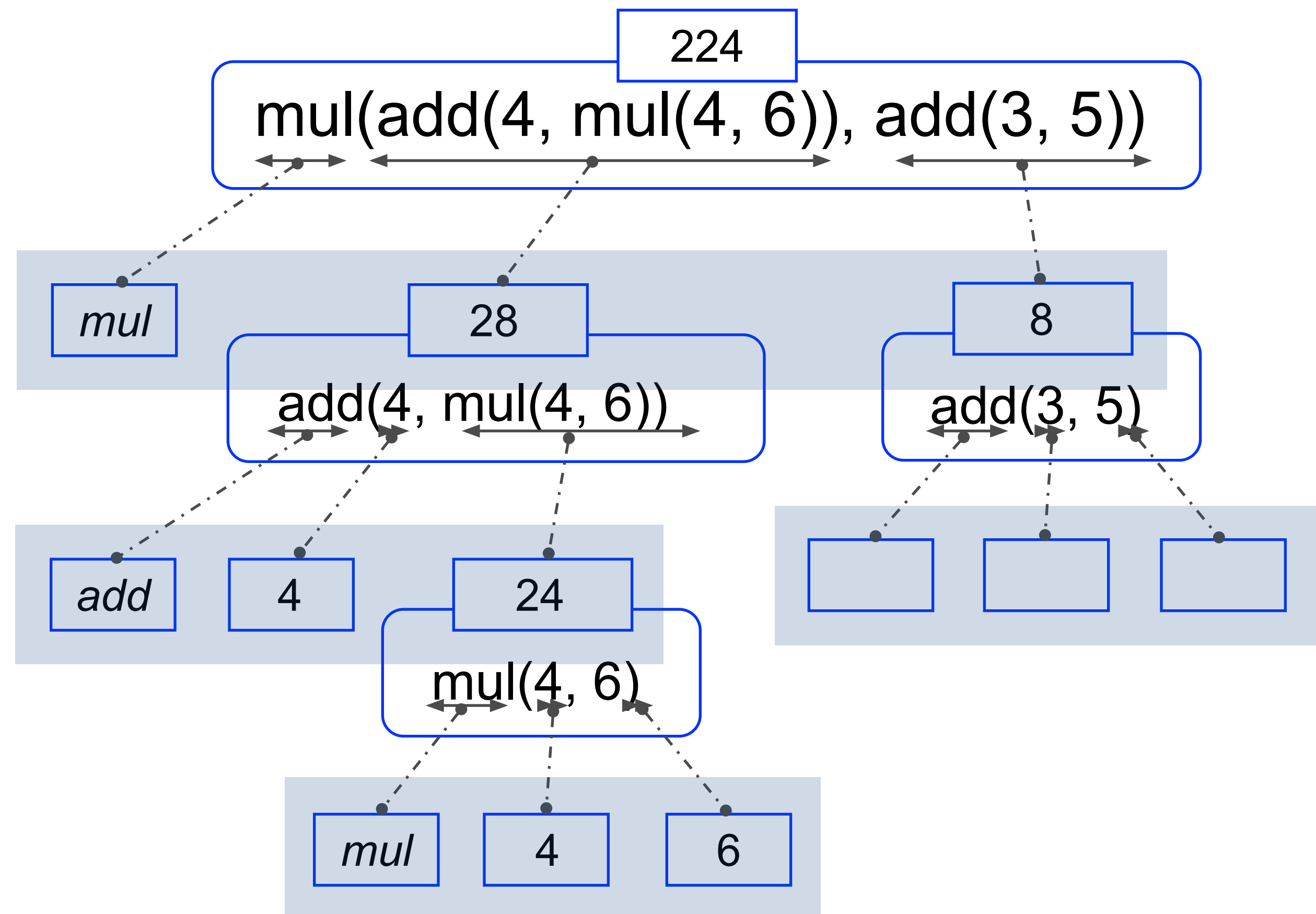
Operators and operands are also expressions

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

