

Functions

Welcome to CS 61A!

Welcome to CS 61A!

John DeNero

denero@berkeley.edu

Welcome to CS 61A!

John DeNero

denero@berkeley.edu

Office hours in 781 Soda:

Welcome to CS 61A!

John DeNero

denero@berkeley.edu

Office hours in 781 Soda:

10:30–11:30 Wednesday (starting next week)

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)

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)

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)

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`

Welcome to CS 61A!

John DeNero
denero@berkeley.edu

Office hours in 781 Soda:

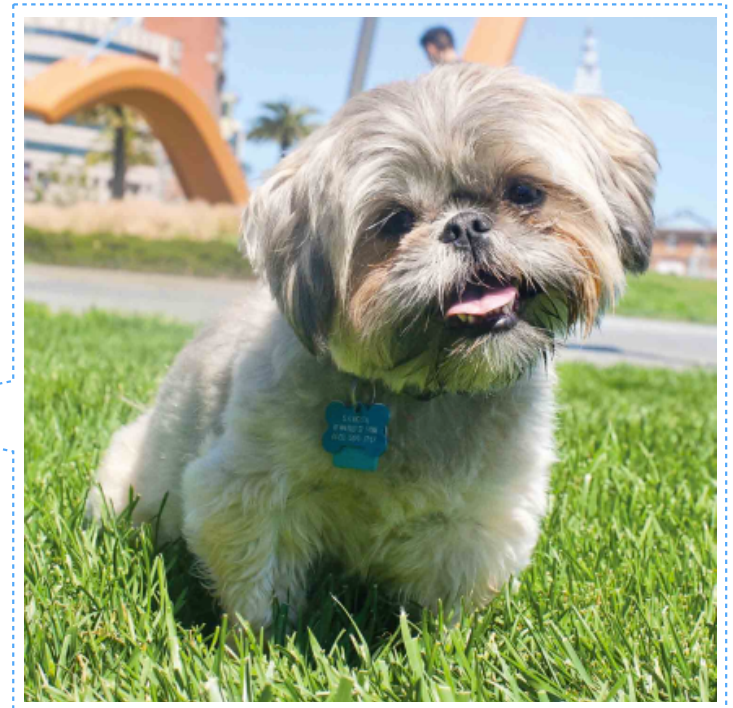
10:30–11:30 Wednesday (starting next week)

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



Welcome to CS 61A!

John DeNero
denero@berkeley.edu

Office hours in 781 Soda:

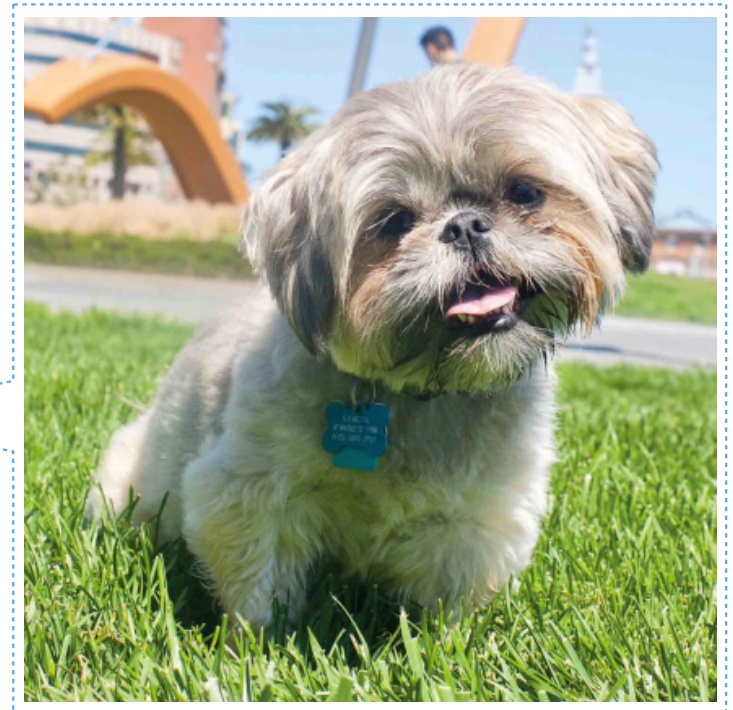
10:30–11:30 Wednesday (starting next week)

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

Welcome to CS 61A!

John DeNero
denero@berkeley.edu

Office hours in 781 Soda:

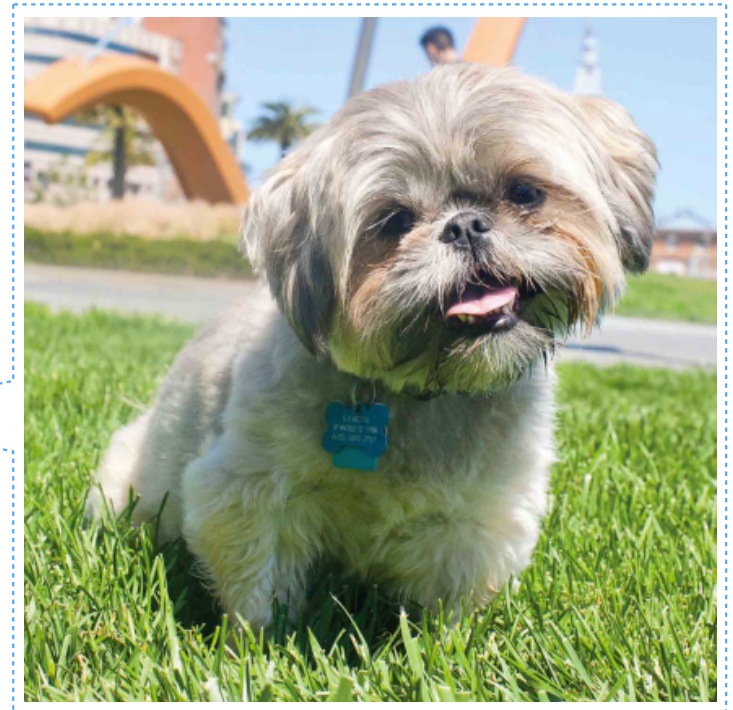
10:30–11:30 Wednesday (starting next week)

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

The 61A Community

52 teaching assistants (TAs), formally known at Berkeley as GSIs or UGSIs:

The 61A Community

52 teaching assistants (TAs), formally known at Berkeley as GSIs or UGSIs:

- Teach lab & discussion sections

The 61A Community

52 teaching assistants (TAs), formally known at Berkeley as GSIs or UGSIs:

- Teach lab & discussion sections
- Hold drop-in office hours

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.

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:

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

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

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.

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

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

Parts of the Course

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

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*)

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*)

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*)

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>

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

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

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?

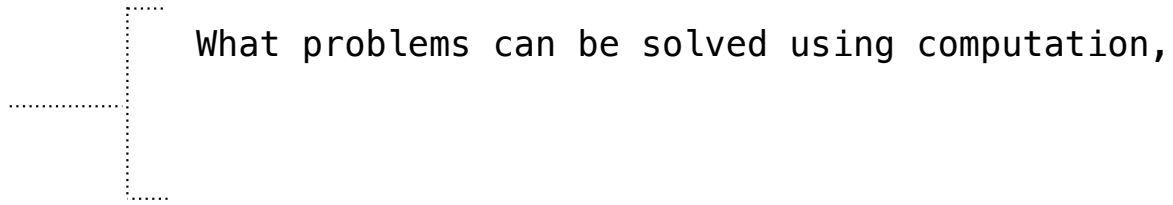
What is Computer Science?

The study of

What is Computer Science?

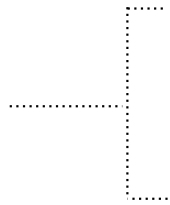
The study of

What problems can be solved using computation,



What is Computer Science?

The study of



- What problems can be solved using computation,
- How to solve those problems, and

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

What is Computer Science?

The study of

Systems

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

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

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

Graphics

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

Graphics

Security

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

Graphics

Security

Networking

Programming Languages

Theory

Scientific Computing

...

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

Graphics

Security

Networking

Programming Languages

Theory

Scientific Computing

...

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

Graphics

Security

Networking

Programming Languages

Theory

Scientific Computing

...

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

Graphics

- Robotics

Security

Networking

Programming Languages

Theory

Scientific Computing

...

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

Graphics

Security

Networking

Programming Languages

Theory

Scientific Computing

...

Decision Making

Robotics

Natural Language Processing

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

Graphics

Security

Networking

Programming Languages

Theory

Scientific Computing

...

Decision Making

Robotics

Natural Language Processing

...

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

Graphics

Security

Networking

Programming Languages

Theory

Scientific Computing

...

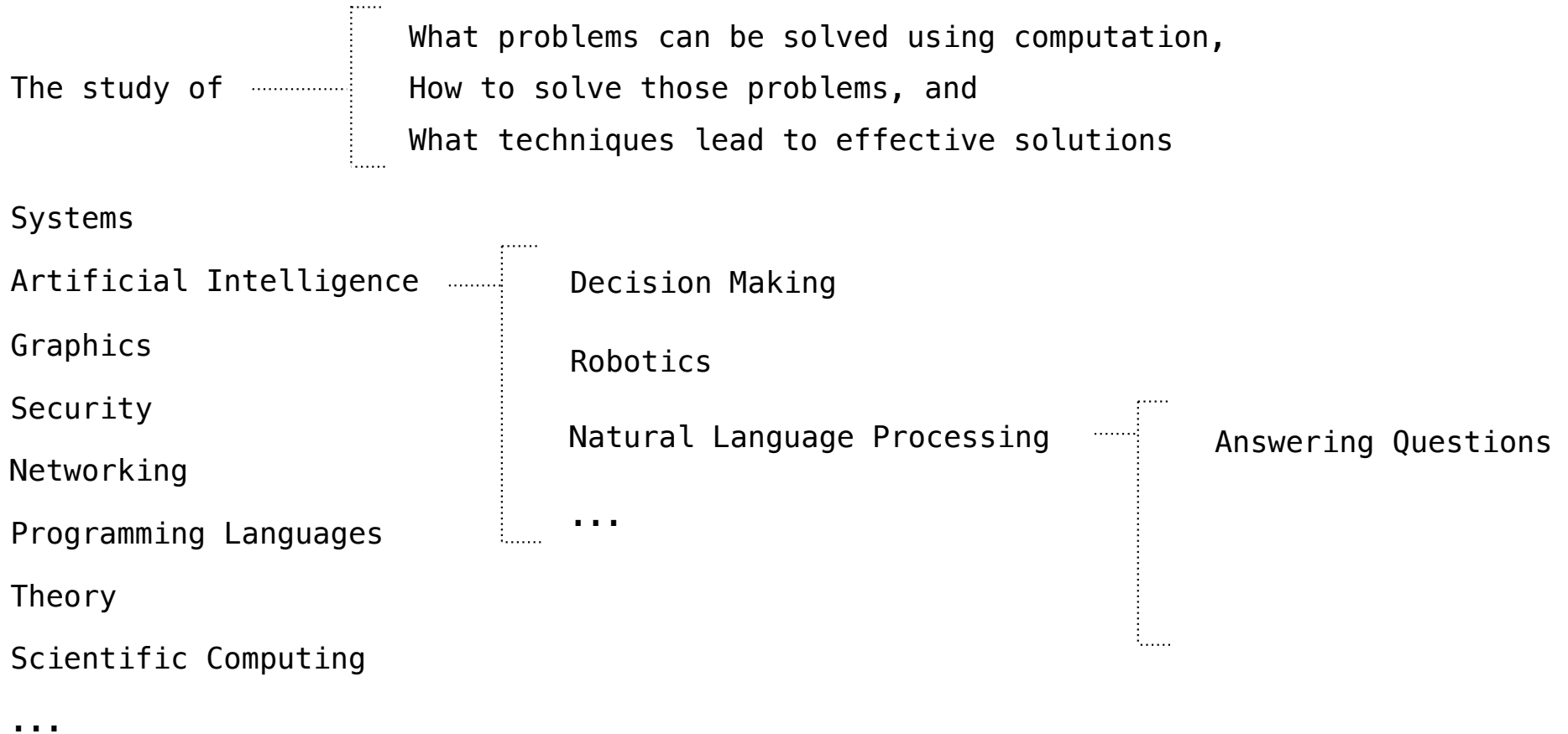
Decision Making

Robotics

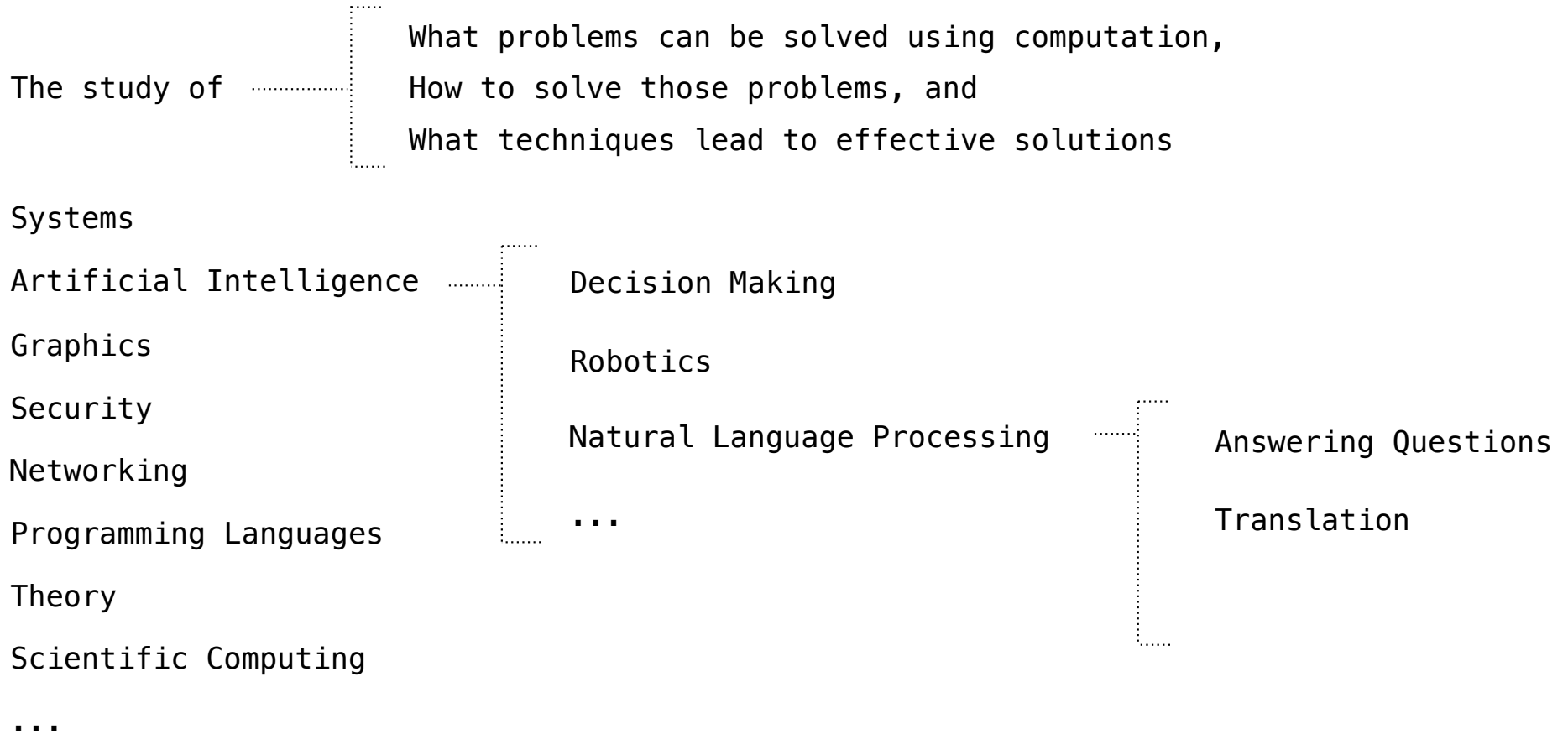
Natural Language Processing

...

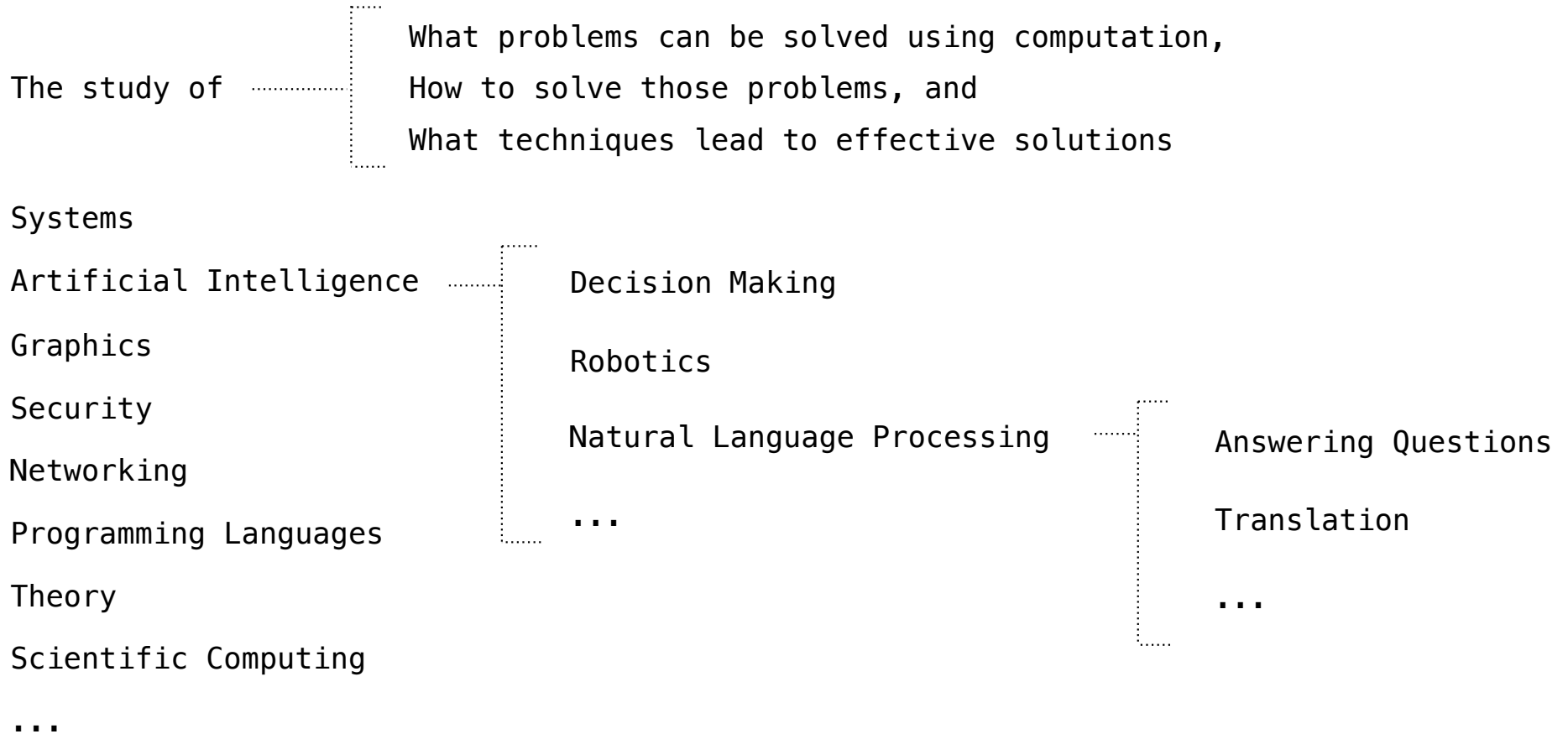
What is Computer Science?



What is Computer Science?



What is Computer Science?



What is This Course About?

What is This Course About?

A course about managing complexity

What is This Course About?

A course about managing complexity

Mastering abstraction

What is This Course About?

A course about managing complexity

Mastering abstraction

Programming paradigms

What is This Course About?

A course about managing complexity

Mastering abstraction

Programming paradigms

An introduction to programming

What is This Course About?

A course about managing complexity

Mastering abstraction

Programming paradigms

An introduction to programming

Full understanding of Python fundamentals



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



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



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



λ



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

CS 10: The Beauty and Joy of Computing



CS 10: The Beauty and Joy of Computing



CS 10: The Beauty and Joy of Computing

Designed for students without prior experience



CS 10: The Beauty and Joy of Computing

Designed for students without prior experience

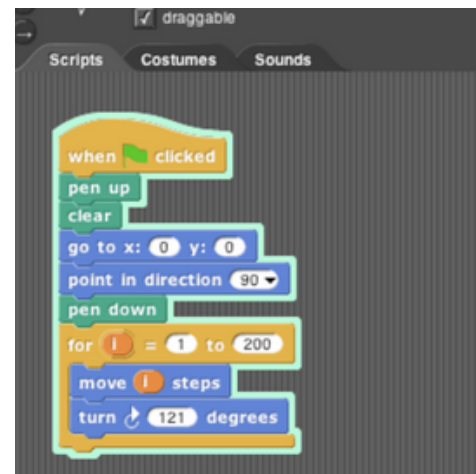
A programming environment created by Berkeley,
now used in courses around the world and online



CS 10: The Beauty and Joy of Computing

Designed for students without prior experience

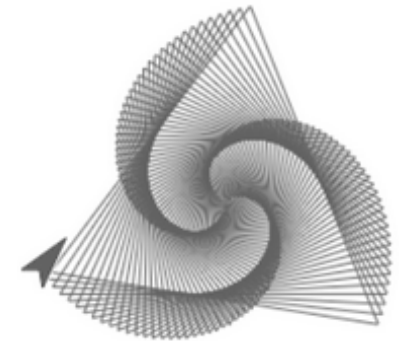
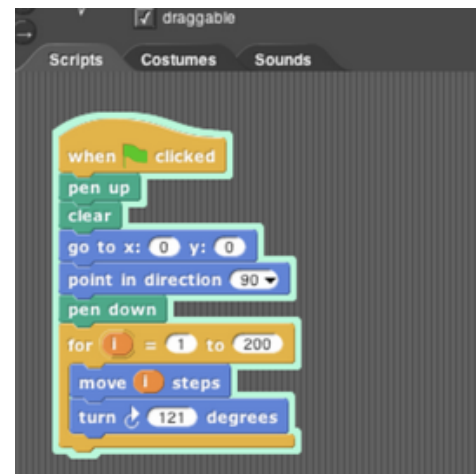
A programming environment created by Berkeley,
now used in courses around the world and online



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

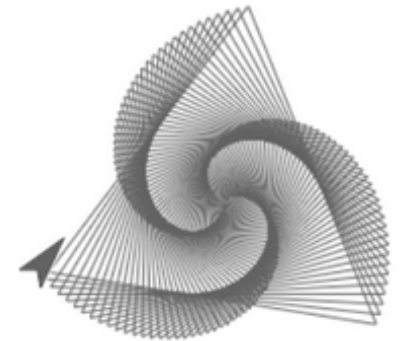
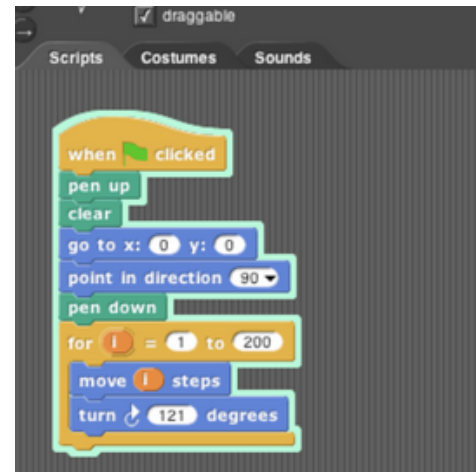


CS 10: The Beauty and Joy of Computing

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



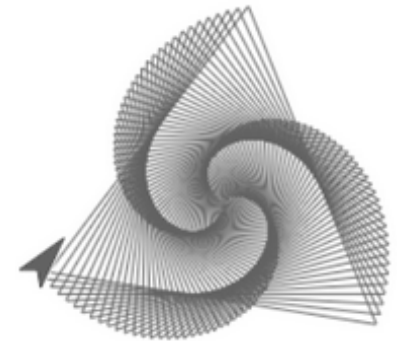
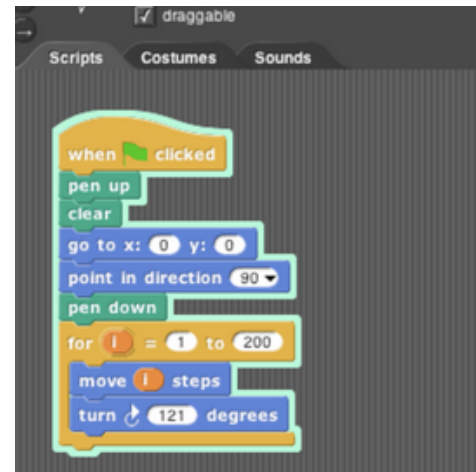
CS 10: The Beauty and Joy of Computing

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



CS 10: The Beauty and Joy of Computing

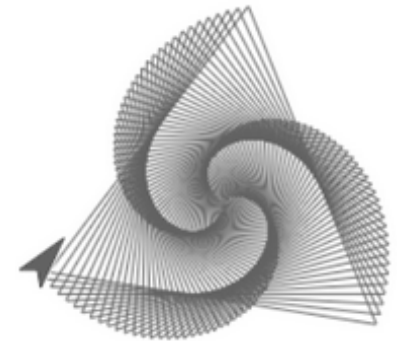
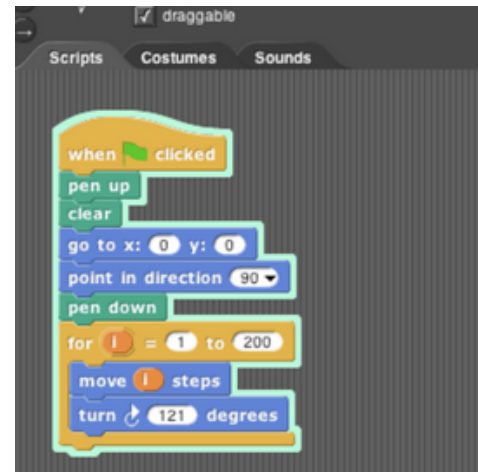
Designed for students without prior experience

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



CS 10: The Beauty and Joy of Computing

Designed for students without prior experience

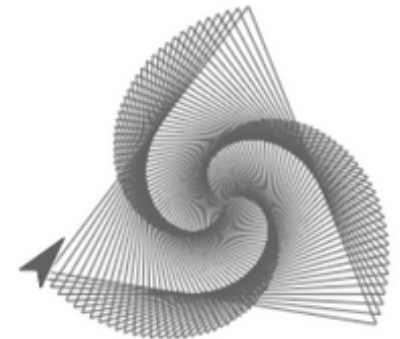
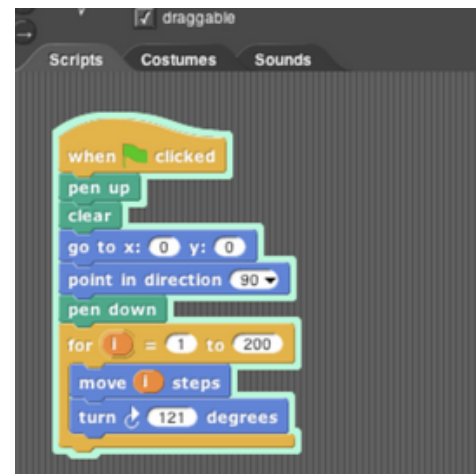
A programming environment created by Berkeley,
now used in courses around the world and online

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/>



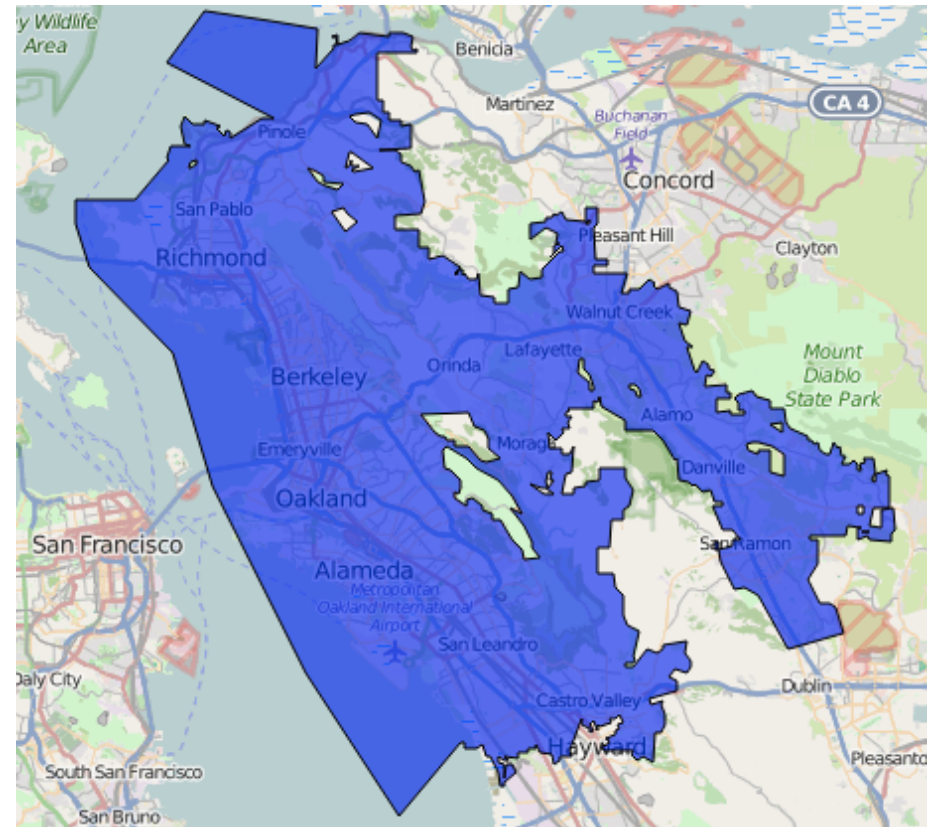
Data Science 8: Foundations of Data Science

Data Science 8: Foundations of Data Science

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

Data Science 8: Foundations of Data Science

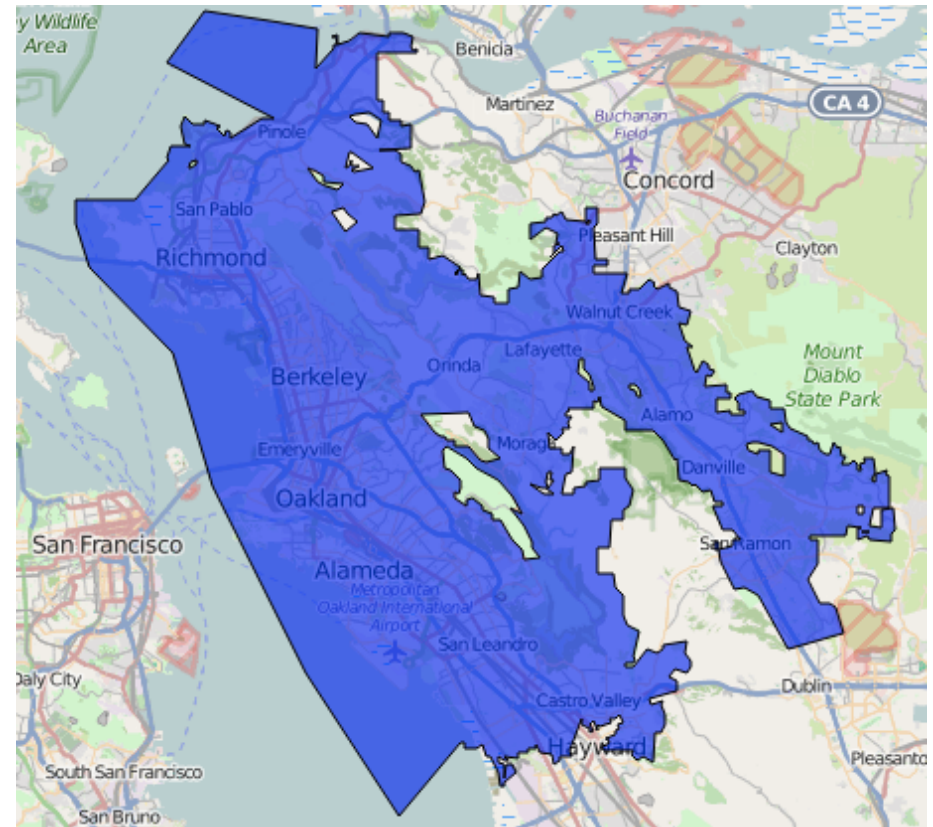
Fundamentals of computing, statistical inference, & machine learning applied to real-world data sets



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

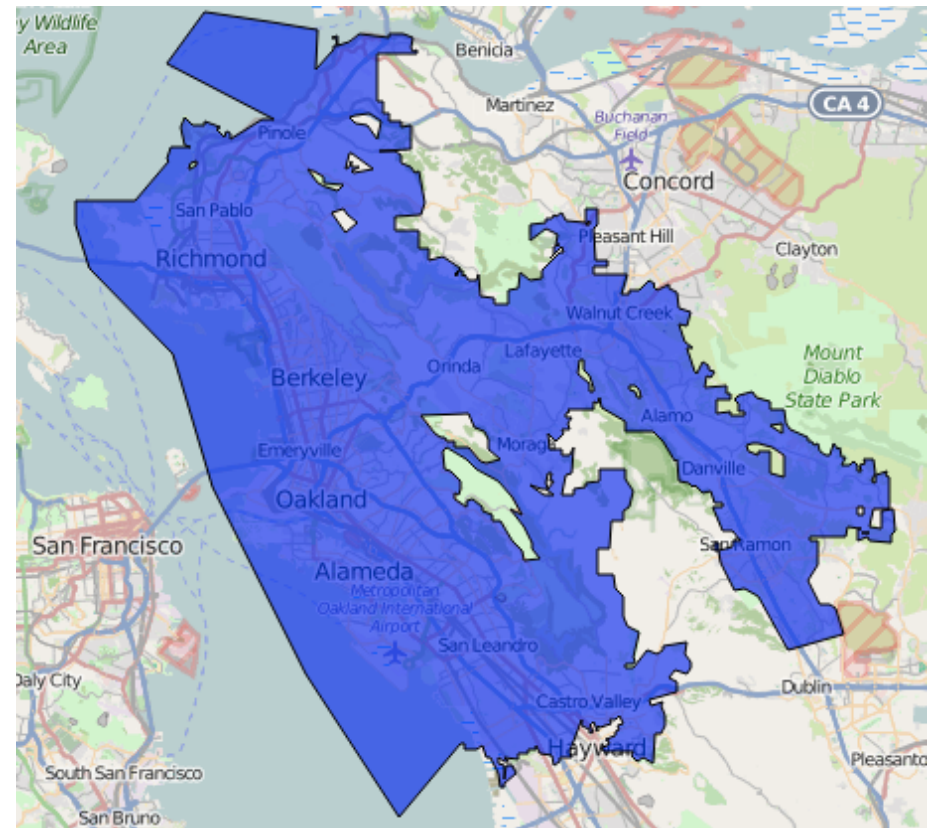


Data Science 8: Foundations of Data Science

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

More statistics than computer science

Great programming practice for CS 61A



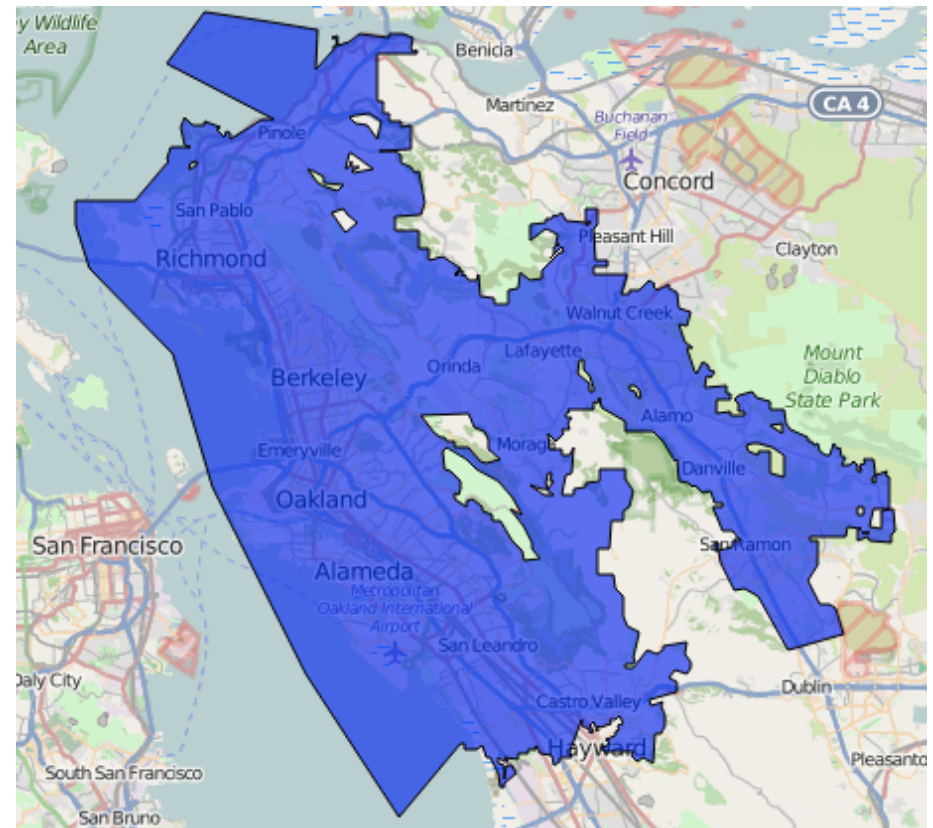
Data Science 8: Foundations of Data Science

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



Data Science 8: Foundations of Data Science

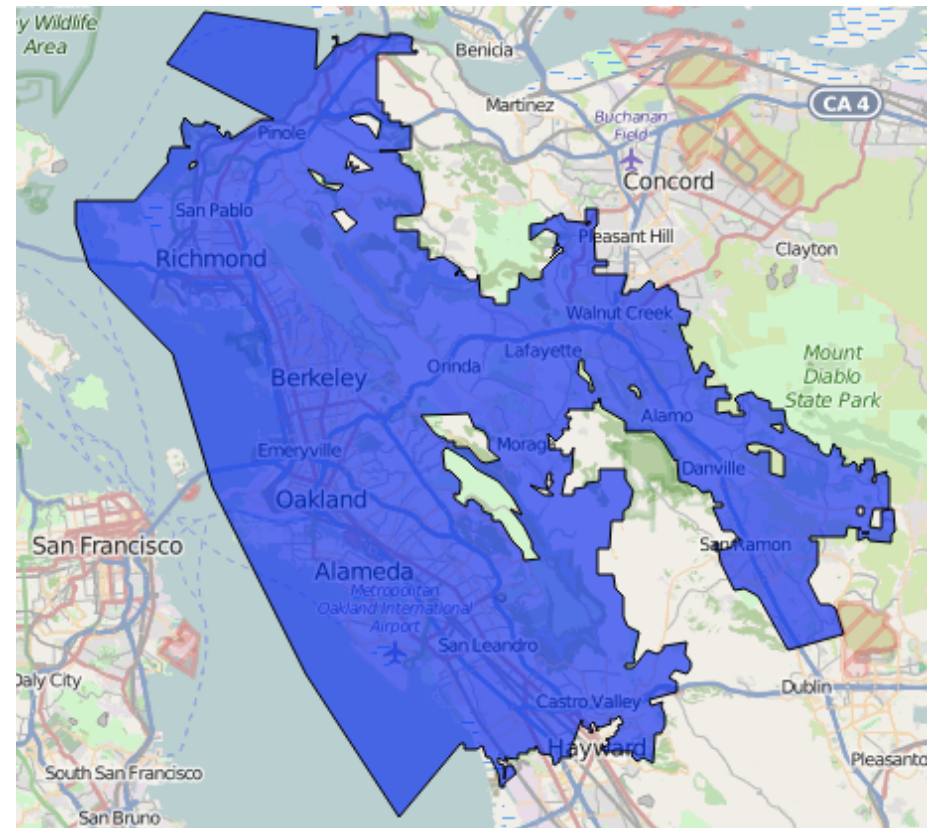
Fundamentals of computing, statistical inference, & machine learning applied to real-world data sets

More statistics than computer science

Great programming practice for CS 61A

Listed as CS C8

180+ person waitlist



Data Science 8: Foundations of Data Science

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

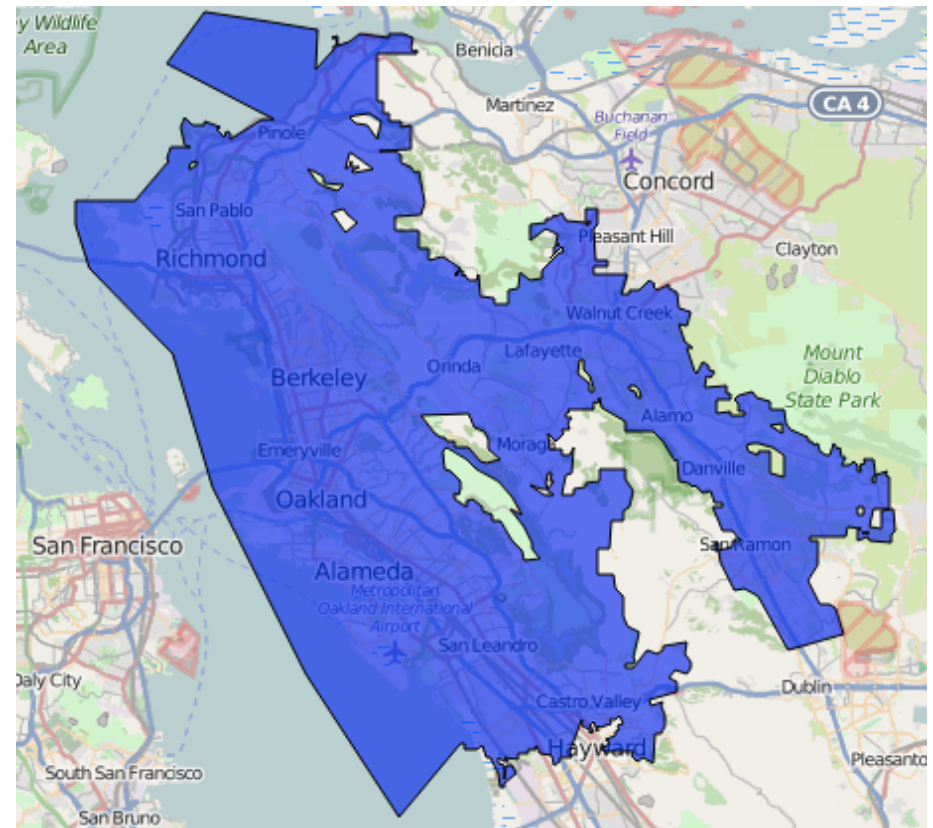
More statistics than computer science

Great programming practice for CS 61A

Listed as CS C8

180+ person waitlist

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



Course Policies

Course Policies

Learning

Learning
Community

Learning

Community

Course Staff

Learning
Community
Course Staff

Details...

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

Collaboration

Collaboration

Asking questions is highly encouraged

Collaboration

Asking questions is highly encouraged

- Discuss everything with each other; learn from your fellow students!

Collaboration

Asking questions is highly encouraged

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

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

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

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

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

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...

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

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

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

Types of expressions

An expression describes a computation and evaluates to a value

Types of expressions

An expression describes a computation and evaluates to a value

$$18 + 69$$

Types of expressions

An expression describes a computation and evaluates to a value

$$18 + 69$$

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

Types of expressions

An expression describes a computation and evaluates to a value

$$18 + 69$$

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

$$\sqrt{3493161}$$

Types of expressions

An expression describes a computation and evaluates to a value

$$18 + 69$$

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

$$\sin \pi$$

$$\sqrt{3493161}$$

Types of expressions

An expression describes a computation and evaluates to a value

$$18 + 69$$

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

$$\sin \pi$$

$$\sqrt{3493161}$$

$$| - 1869 |$$

Types of expressions

An expression describes a computation and evaluates to a value

$$18 + 69$$

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

$$\sin \pi$$

$$\sqrt{3493161}$$

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

$$|-1869|$$

Types of expressions

An expression describes a computation and evaluates to a value

$$18 + 69$$

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

$$\sin \pi$$

$$\sqrt{3493161}$$

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

$$|-1869|$$

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

Types of expressions

An expression describes a computation and evaluates to a value

$$18 + 69$$

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

$$\sin \pi$$

$$f(x)$$

$$\sqrt{3493161}$$

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

$$|-1869|$$

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

Types of expressions

An expression describes a computation and evaluates to a value

$$18 + 69$$

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

$$\sin \pi$$

$$2^{100}$$

$$f(x)$$

$$\sqrt{3493161}$$

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

$$|-1869|$$

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

Types of expressions

An expression describes a computation and evaluates to a value

$$18 + 69$$

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

$$\sin \pi$$

$$\log_2 1024$$

$$2^{100}$$

$$f(x)$$

$$\sqrt{3493161}$$

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

$$|-1869|$$

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

Types of expressions

An expression describes a computation and evaluates to a value

$$18 + 69$$

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

$$\sin \pi$$

$$\log_2 1024$$

$$2^{100}$$

$$f(x)$$

$$\sqrt{3493161}$$

$$7 \bmod 2$$

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

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

$$|-1869|$$

Types of expressions

An expression describes a computation and evaluates to a value

$$18 + 69$$

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

$$\sin \pi$$

$$\log_2 1024$$

$$2^{100}$$

$$f(x)$$

$$\sqrt{3493161}$$

$$7 \bmod 2$$

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

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

$$|-1869|$$

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

Types of expressions

An expression describes a computation and evaluates to a value

$$\begin{array}{ccccccc} 18 + 69 & & \frac{6}{23} & & \sin \pi & & \log_2 1024 \\ 2^{100} & & & & & & \\ 7 \bmod 2 & & \boxed{f(x)} & & \sqrt{3493161} & & \\ | - 1869 | & & \sum_{i=1}^{100} i & & \binom{69}{18} & & \lim_{x \rightarrow \infty} \frac{1}{x} \end{array}$$

Call Expressions in Python

All expressions can use function call notation
(Demo)

Anatomy of a Call Expression

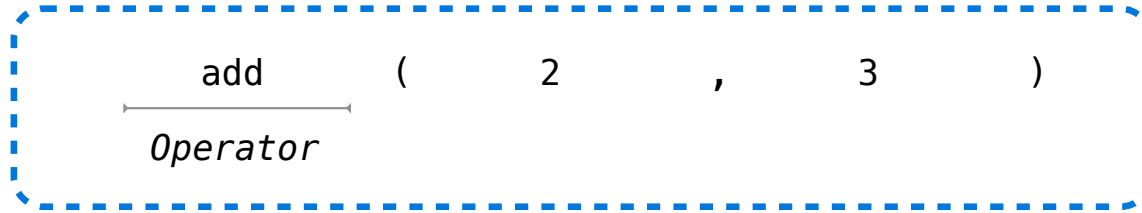
Anatomy of a Call Expression

add (2 , 3)

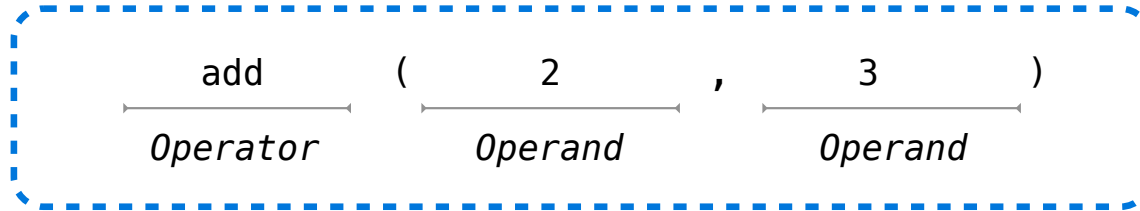
Anatomy of a Call Expression

add (2 , 3)

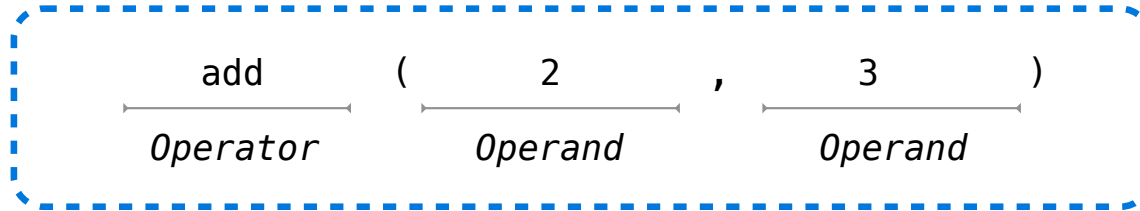
Anatomy of a Call Expression



Anatomy of a Call Expression

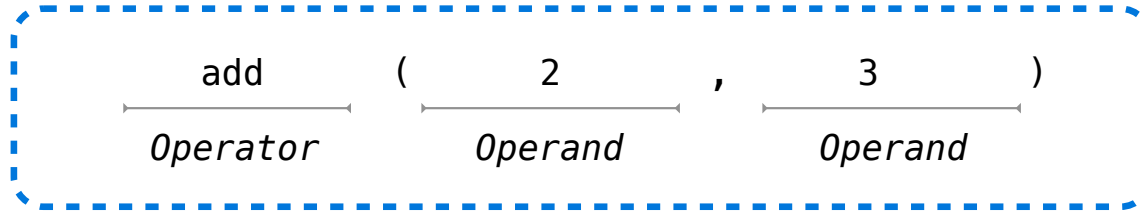


Anatomy of a Call Expression



Operators and operands are also expressions

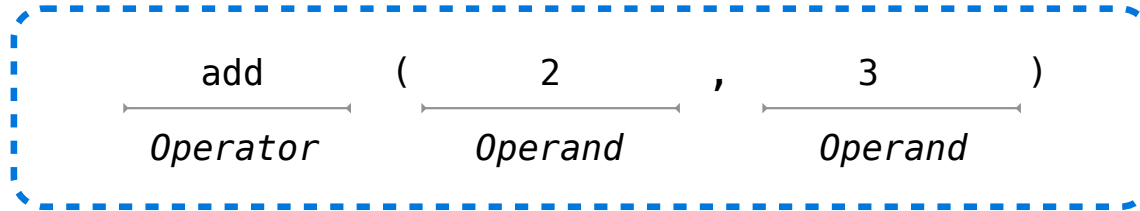
Anatomy of a Call Expression



Operators and operands are also expressions

So they evaluate to values

Anatomy of a Call Expression

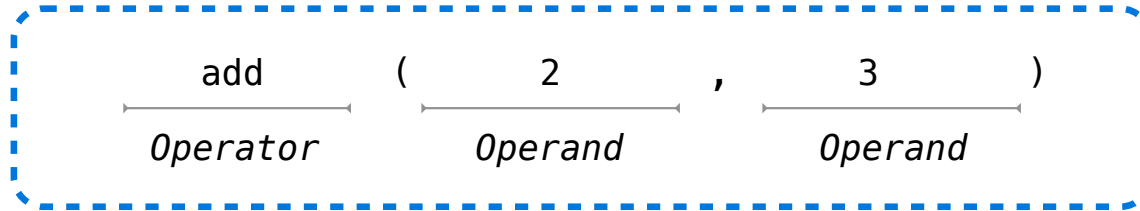


Operators and operands are also expressions

So they evaluate to values

Evaluation procedure for call expressions:

Anatomy of a Call Expression



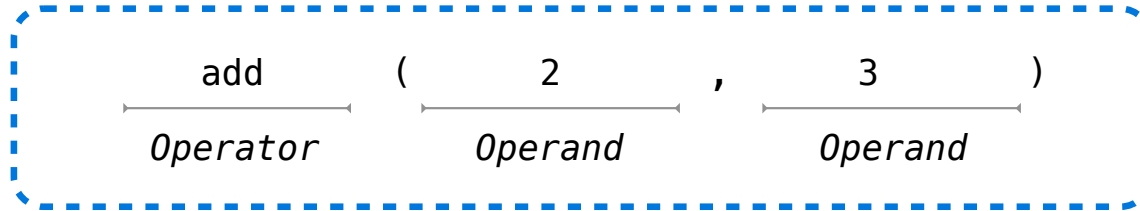
Operators and operands are also expressions

So they evaluate to values

Evaluation procedure for call expressions:

1. Evaluate the operator and then the operand subexpressions

Anatomy of a Call Expression



Operators and operands are also expressions

So they evaluate to values

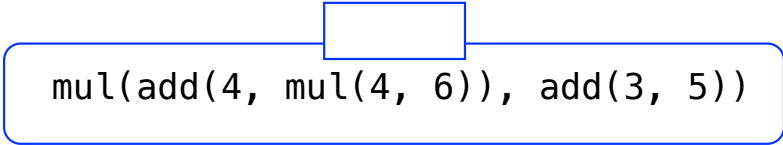
Evaluation procedure for call expressions:

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

```
mul(add(4, mul(4, 6)), add(3, 5))
```

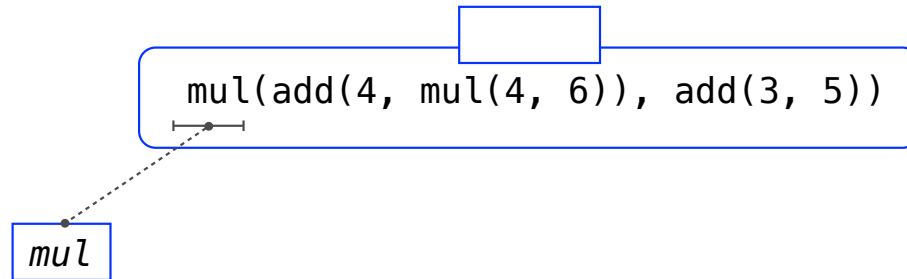

Evaluating Nested Expressions



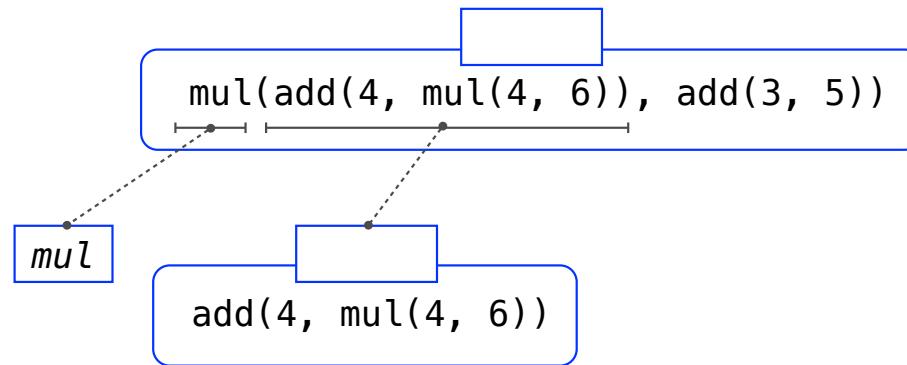
The diagram shows the expression `mul(add(4, mul(4, 6)), add(3, 5))` enclosed in a rounded rectangle. A smaller rectangle is positioned above the `mul(4, 6)` sub-expression, indicating it is the current focus of evaluation.

```
mul(add(4, mul(4, 6)), add(3, 5))
```

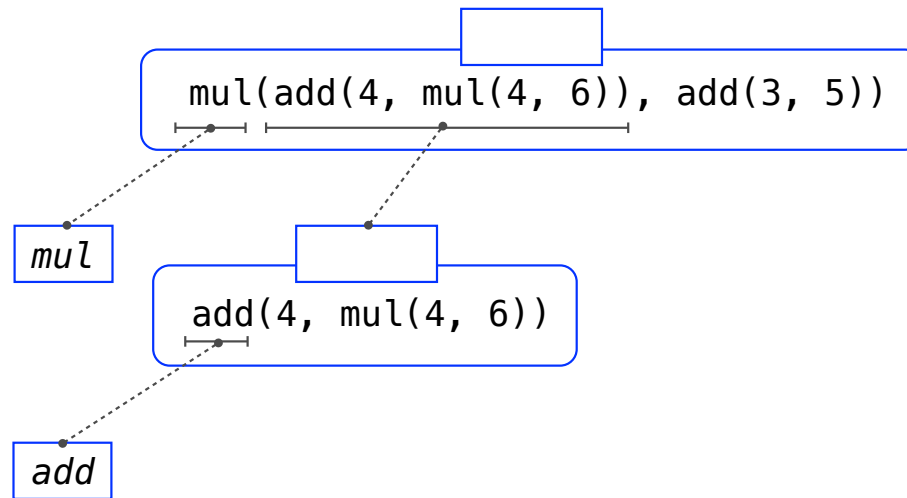
Evaluating Nested Expressions



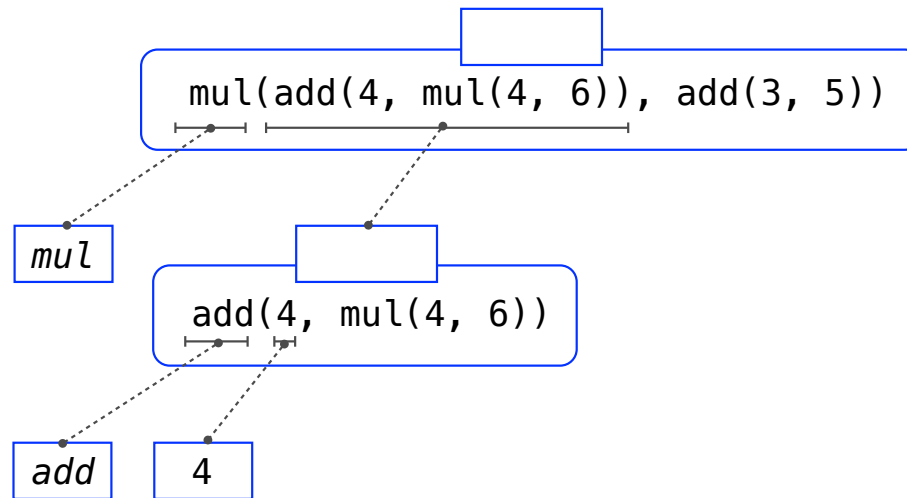
Evaluating Nested Expressions



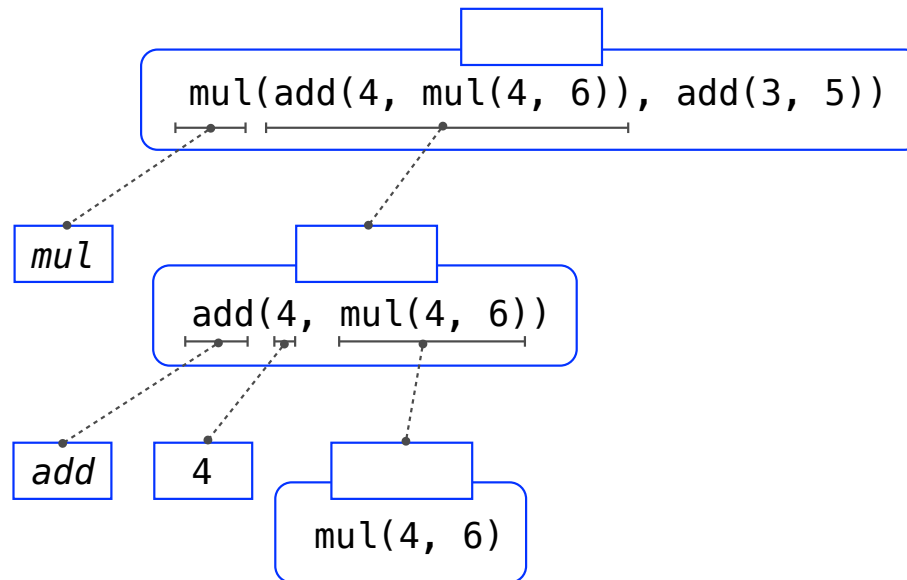
Evaluating Nested Expressions



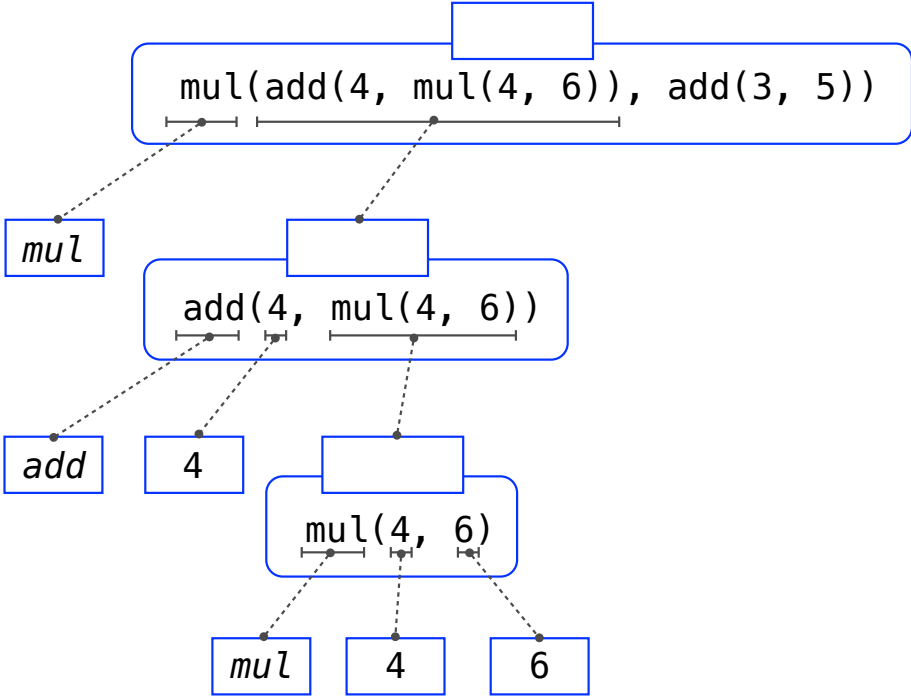
Evaluating Nested Expressions



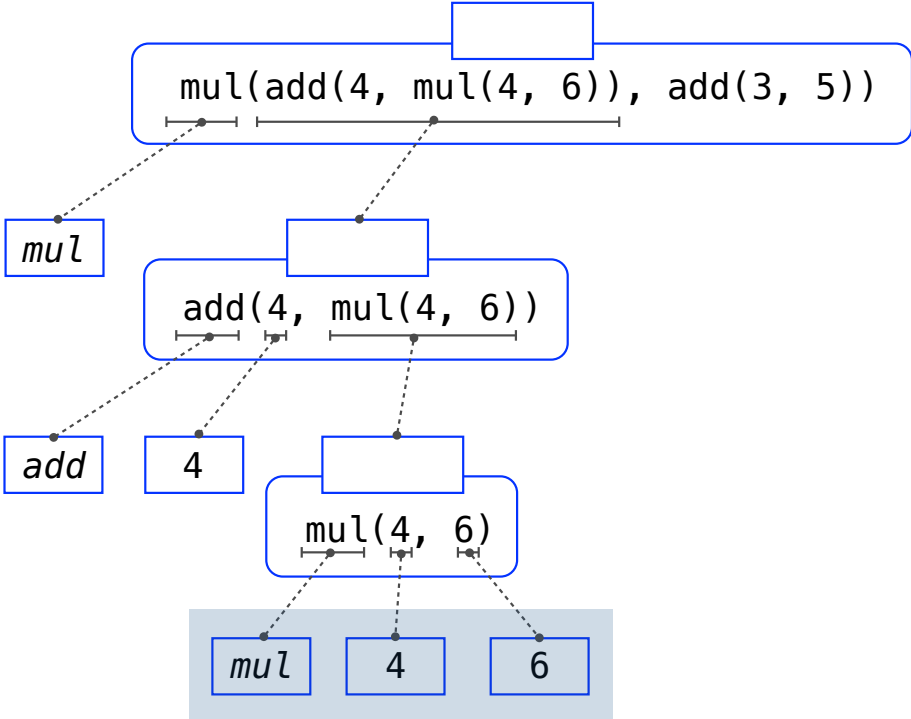
Evaluating Nested Expressions



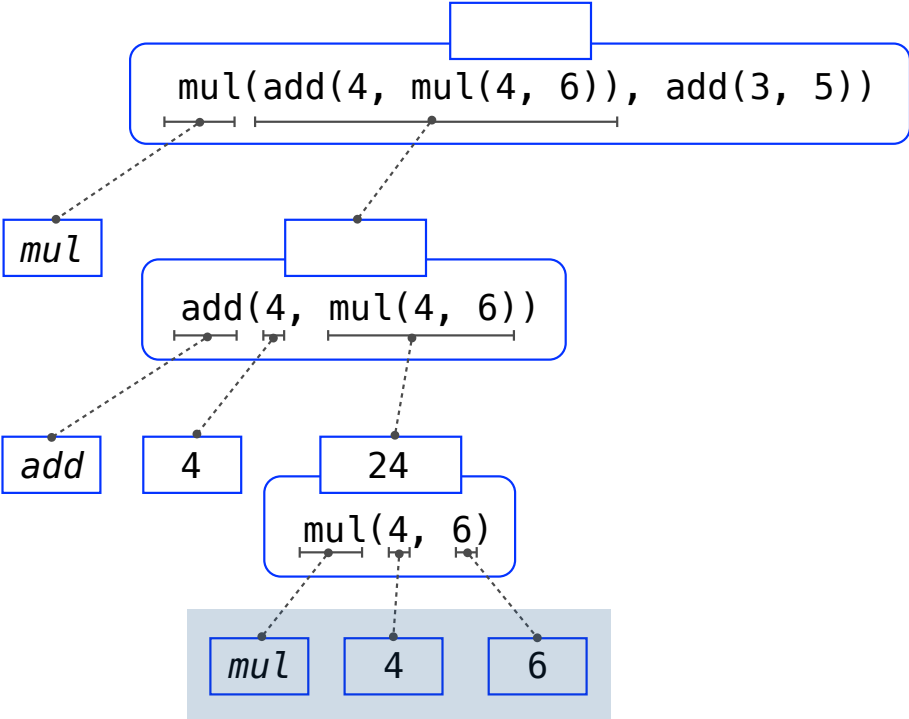
Evaluating Nested Expressions



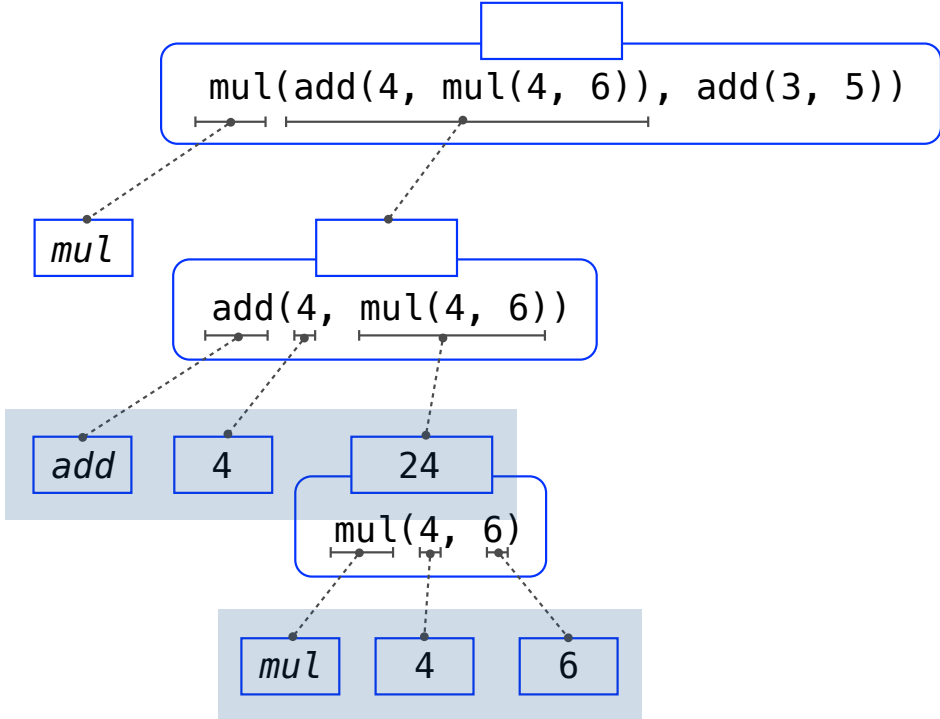
Evaluating Nested Expressions



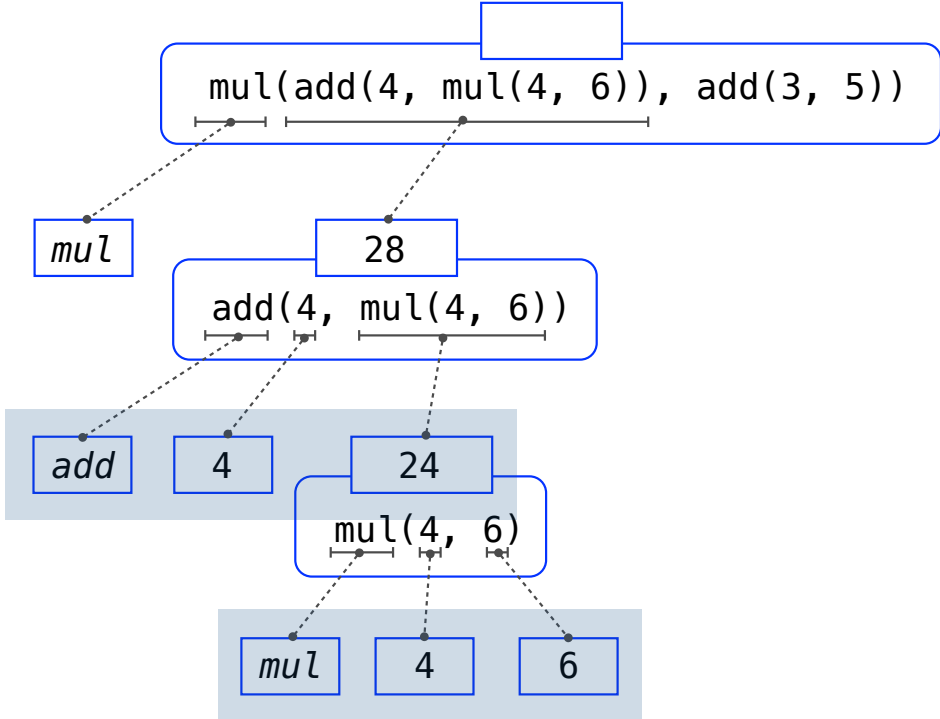
Evaluating Nested Expressions



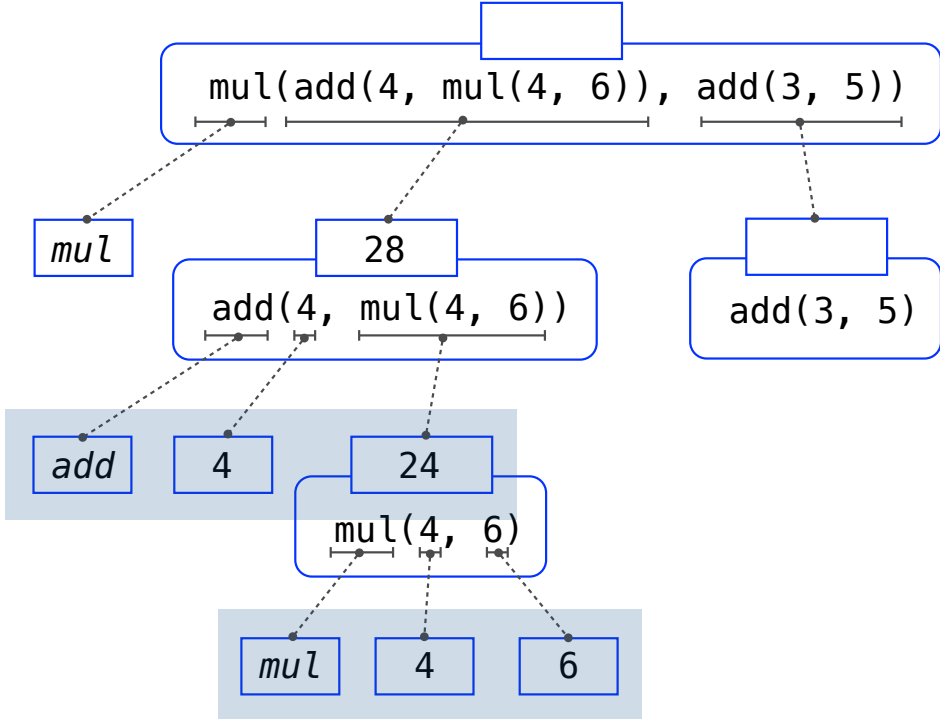
Evaluating Nested Expressions



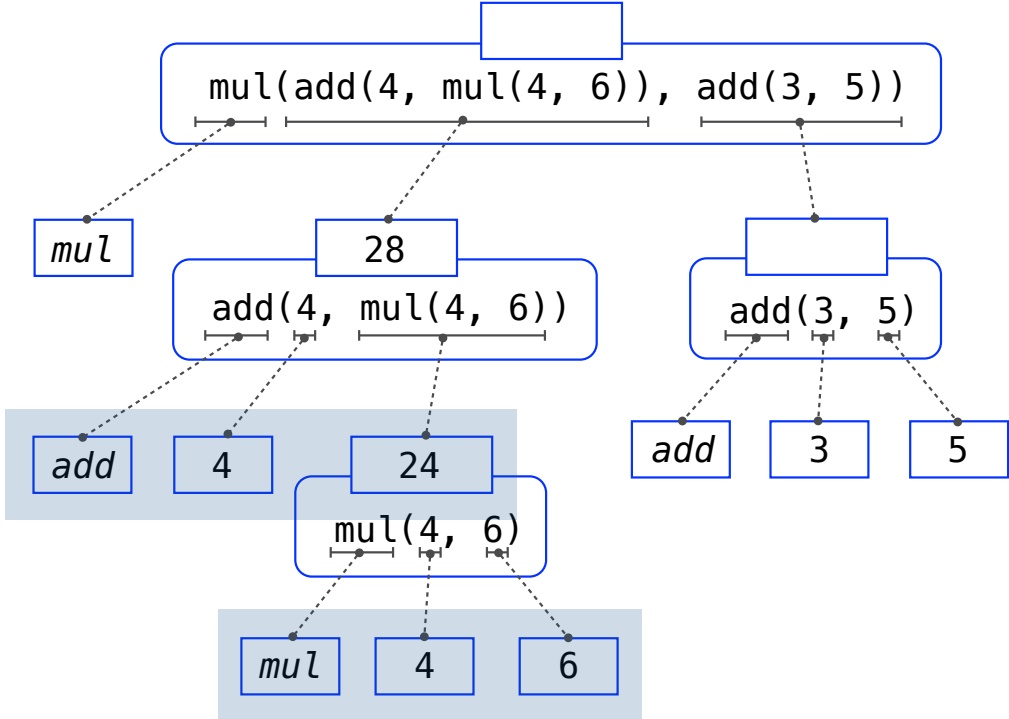
Evaluating Nested Expressions



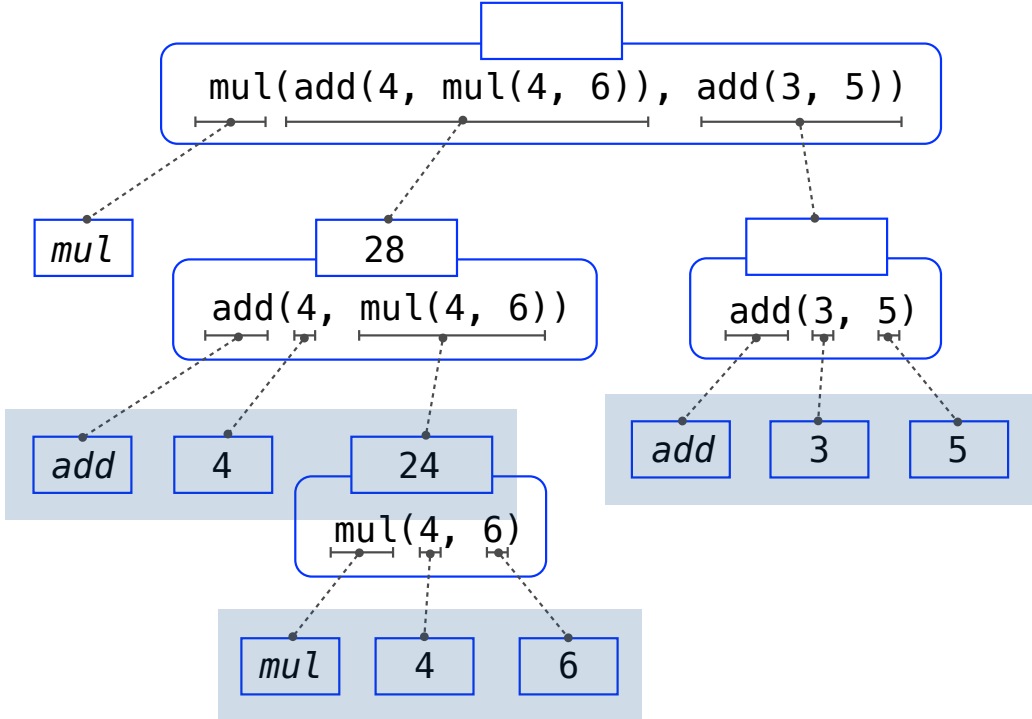
Evaluating Nested Expressions



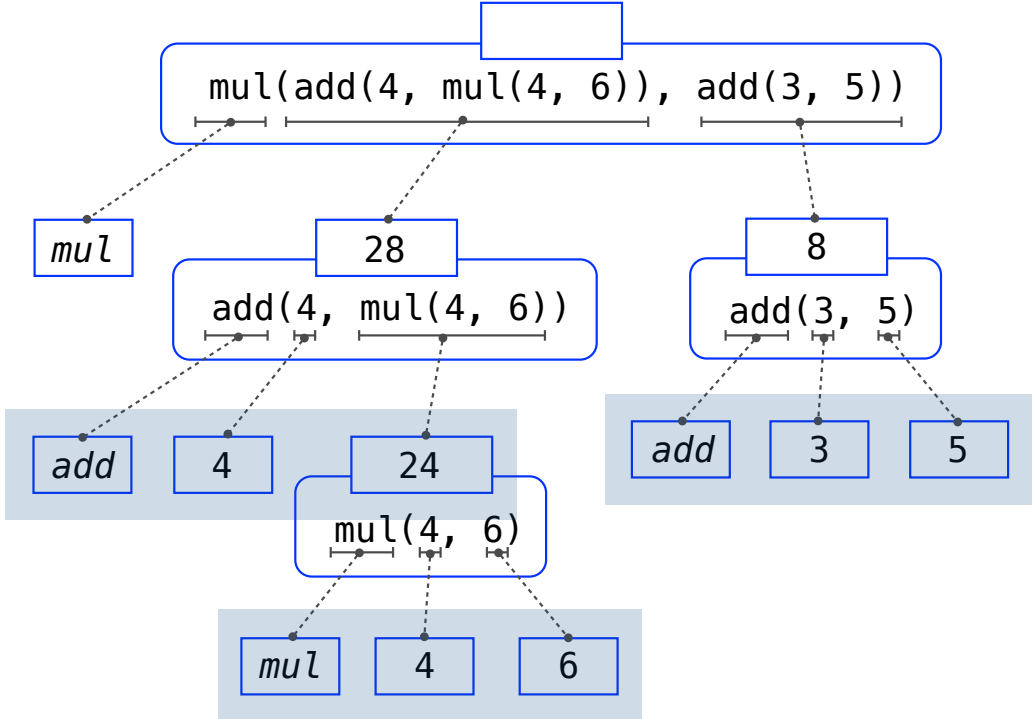
Evaluating Nested Expressions



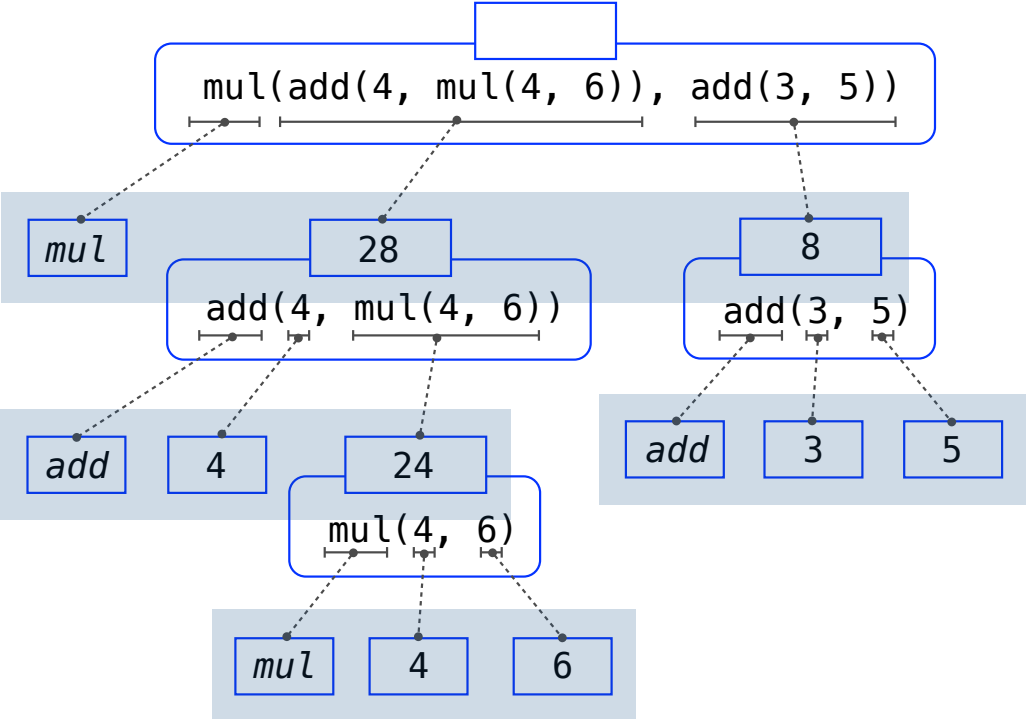
Evaluating Nested Expressions



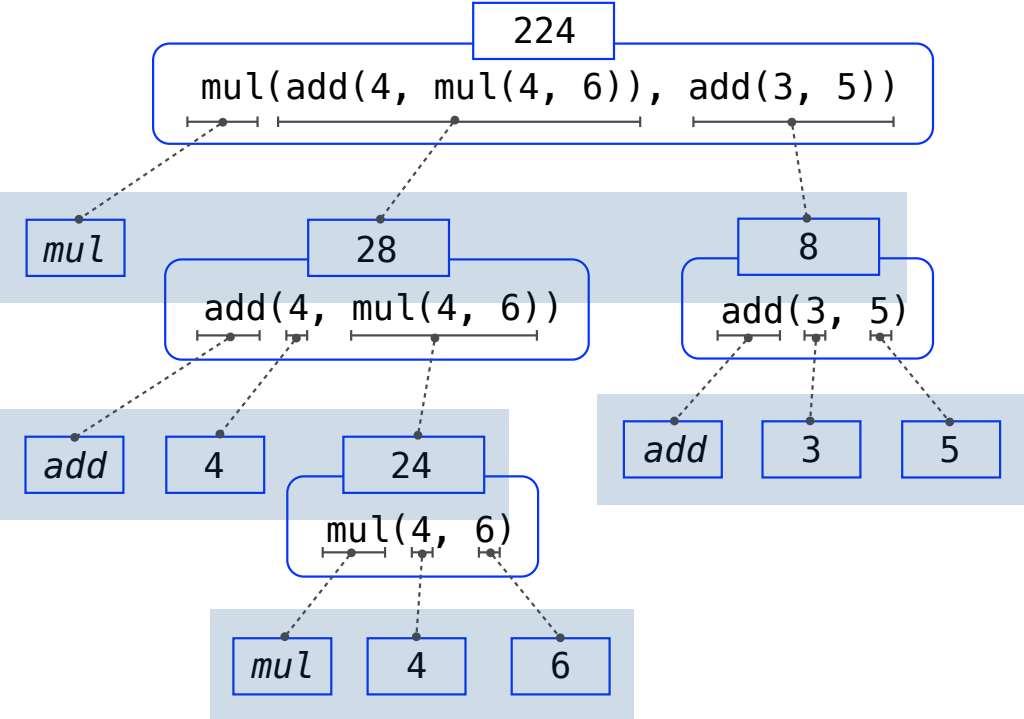
Evaluating Nested Expressions



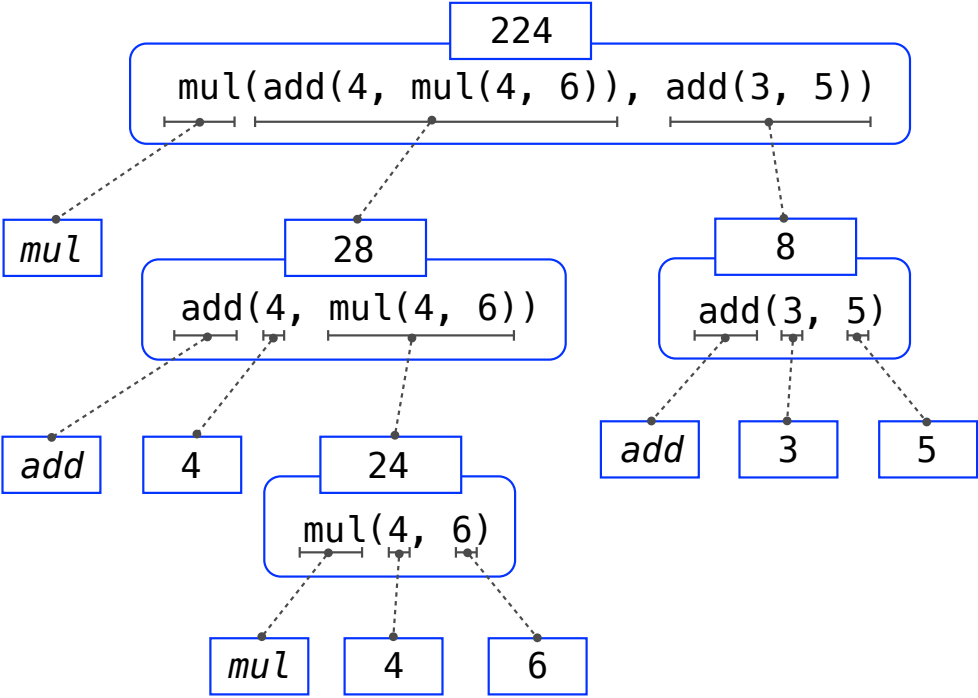
Evaluating Nested Expressions



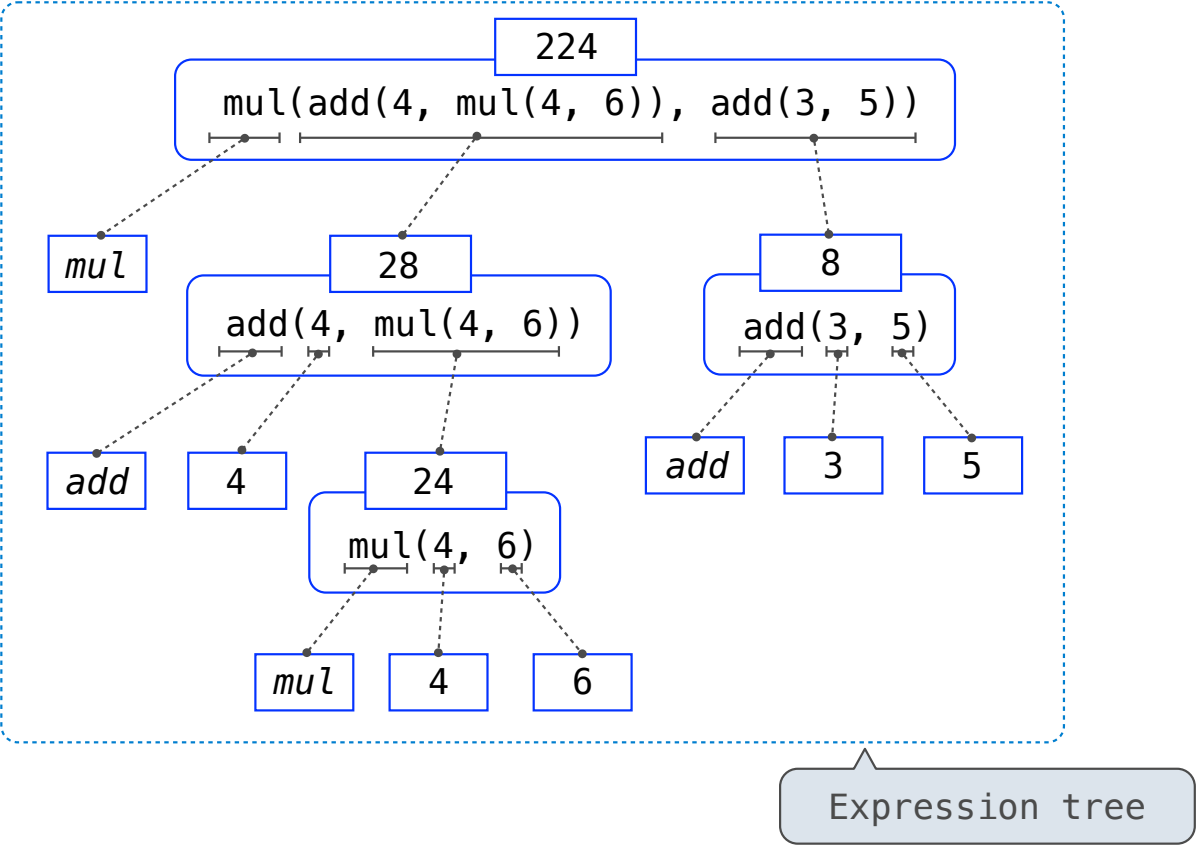
Evaluating Nested Expressions



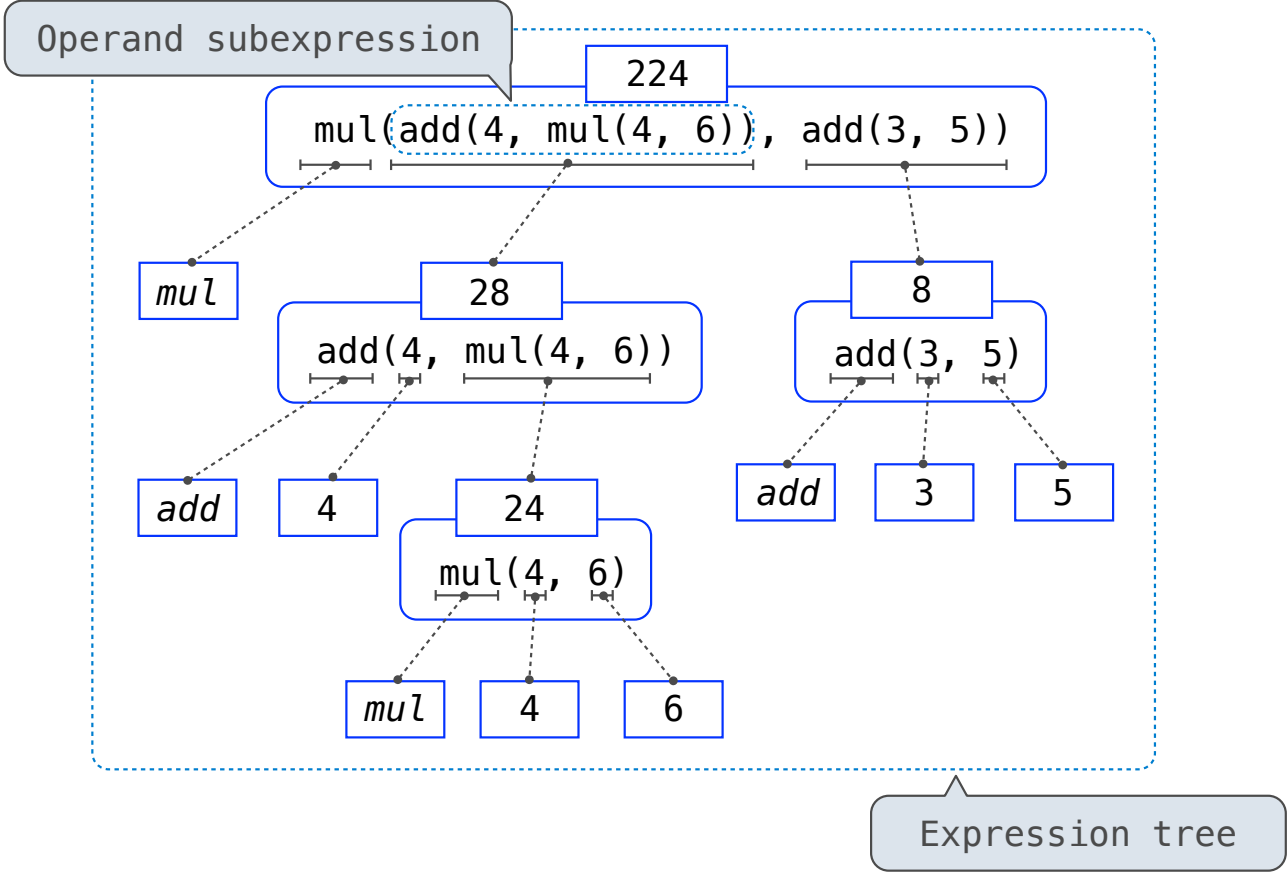
Evaluating Nested Expressions



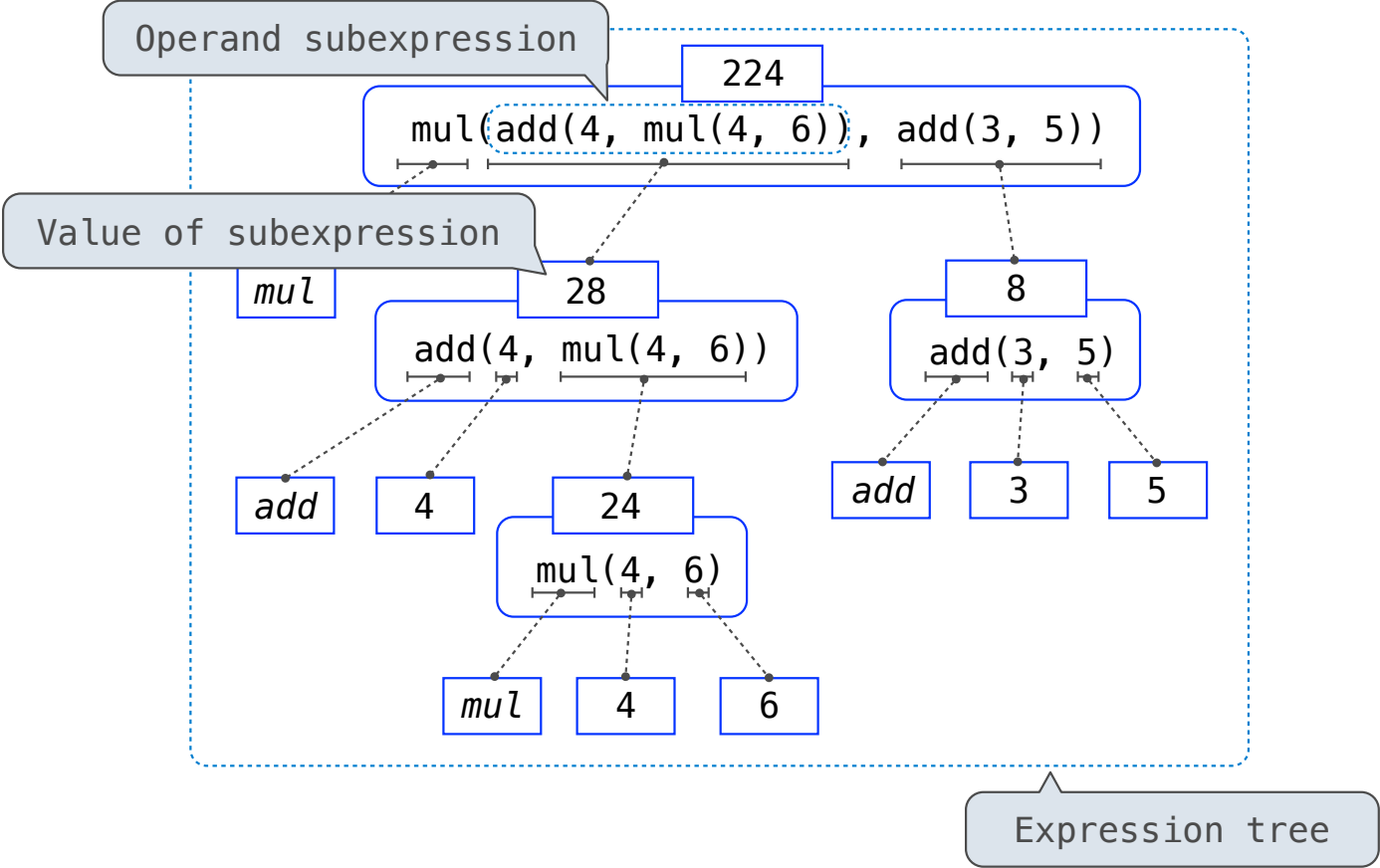
Evaluating Nested Expressions



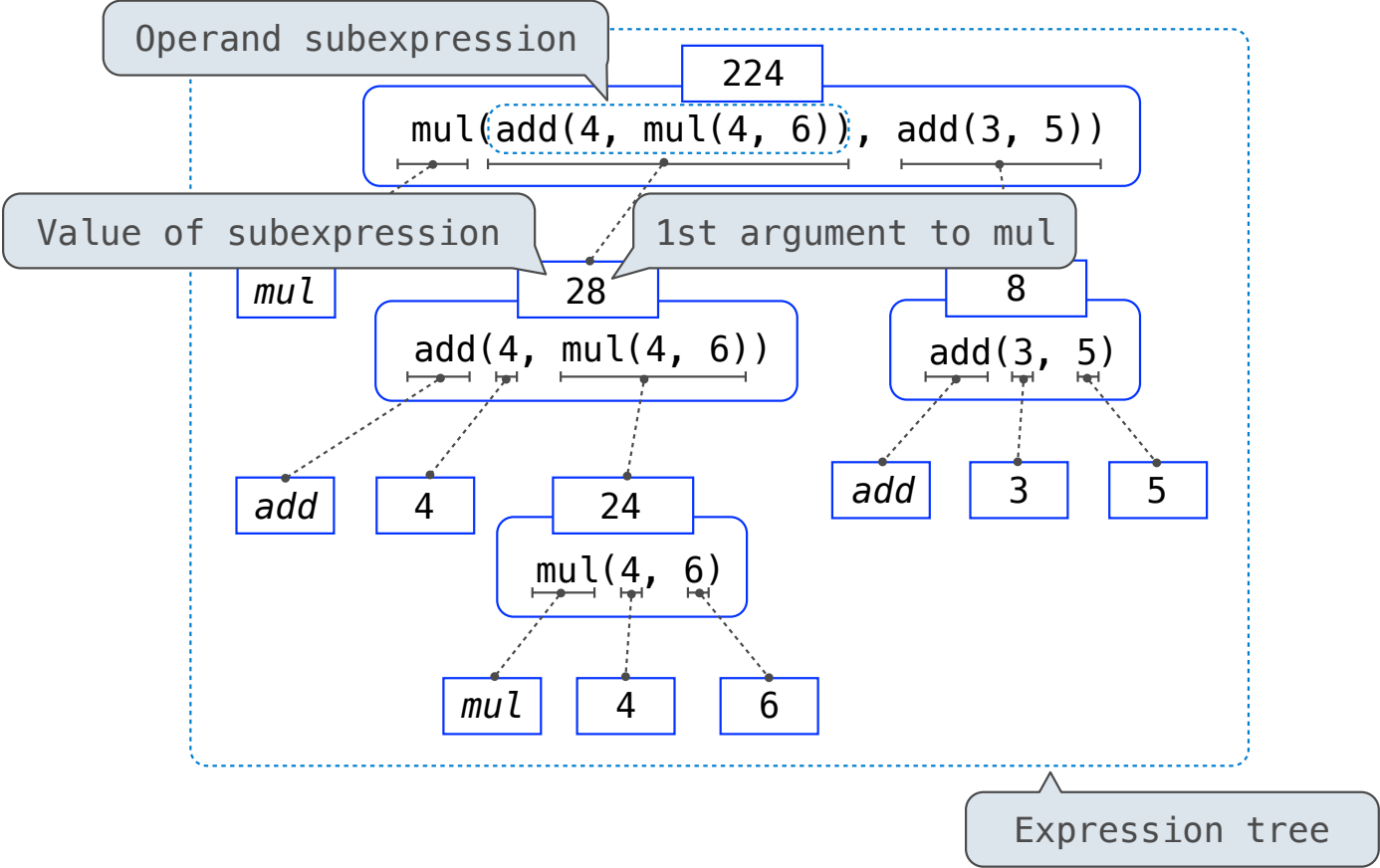
Evaluating Nested Expressions



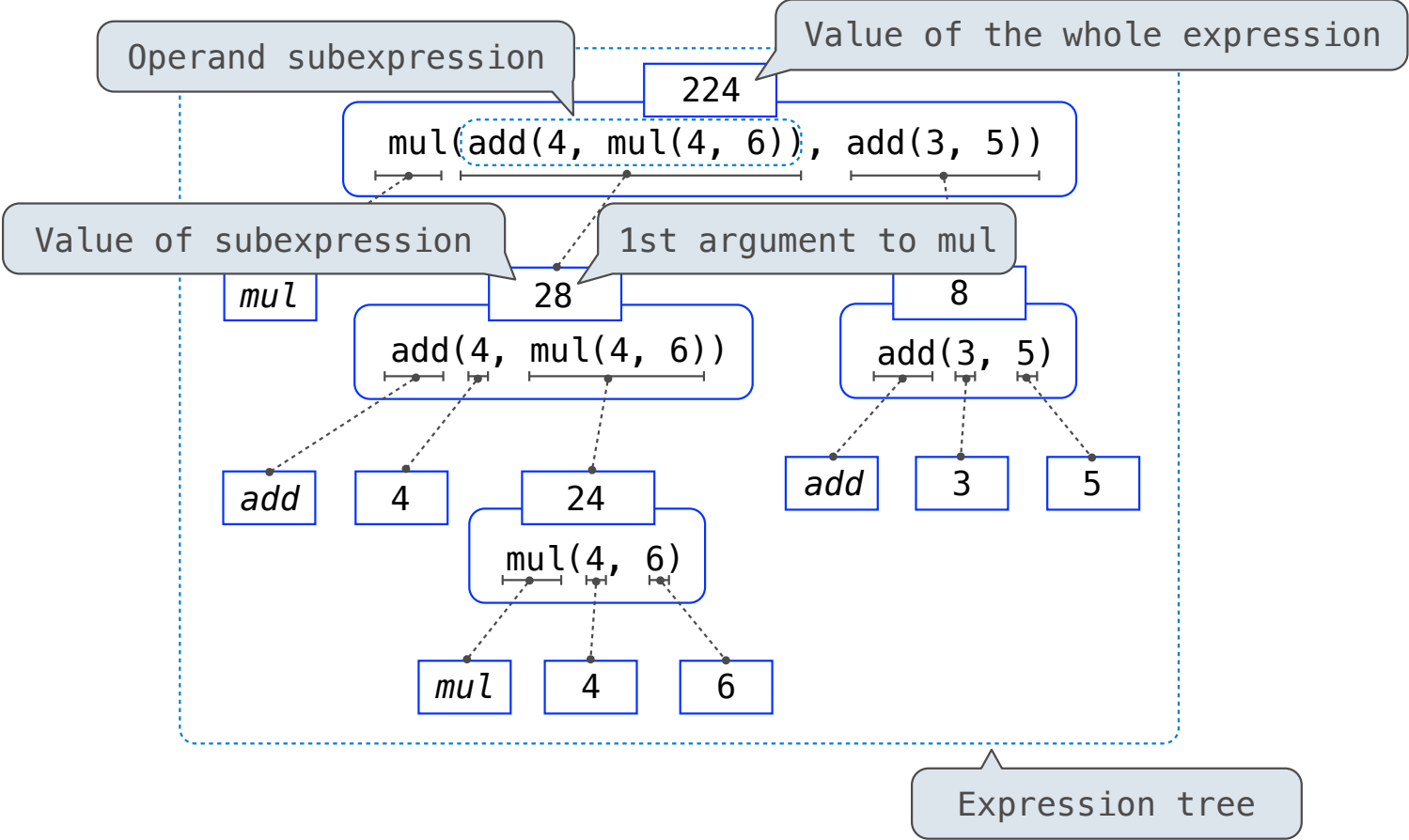
Evaluating Nested Expressions



Evaluating Nested Expressions



Evaluating Nested Expressions



Functions, Values, Objects, Interpreters, and Data

(Demo)