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 (starting next week)
Welcome to CS 61A!

John DeNero
denero@berkeley.edu

Office hours in 781 Soda
(starting next week)
Wed 10am–11am & Thurs 10am–11am
Welcome to CS 61A!

John DeNero
denero@berkeley.edu

Office hours in 781 Soda (starting next week)
Wed 10am–11am & Thurs 10am–11am
Welcome to CS 61A!

John DeNero
denero@berkeley.edu

Office hours in 781 Soda (starting next week)
Wed 10am–11am & Thurs 10am–11am
By appointment: denero.org/meet.html
Welcome to CS 61A!

John DeNero
denero@berkeley.edu

Office hours in 781 Soda (starting next week)
Wed 10am–11am & Thurs 10am–11am
By appointment: denero.org/meet.html

Fastest way to get answers: piazza.com/berkeley/spring2018/cs61a
Welcome to CS 61A!

John DeNero
denero@berkeley.edu

Office hours in 781 Soda
(starting next week)
Wed 10am–11am & Thurs 10am–11am
By appointment: denero.org/meet.html

Fastest way to get answers: piazza.com/berkeley/spring2018/cs61a
Contact me & heads of staff: cs61a@berkeley.edu
The 61A Community
The 61A Community

44 teaching assistants (TAs), formally known at Berkeley as UGSIs:
The 61A Community

44 teaching assistants (TAs), formally known at Berkeley as UGSIs:
• Teach lab & discussion sections
The 61A Community

44 teaching assistants (TAs), formally known at Berkeley as UGSIs:
- Teach lab & discussion sections
- Hold drop-in office hours
The 61A Community

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

• Teach lab & discussion sections
• Hold drop-in office hours
• Lots of other stuff: develop assignments, grade exams, etc.
The 61A Community

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

50+ mentors:
The 61A Community

44 teaching assistants (TAs), formally known at Berkeley as 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

44 teaching assistants (TAs), formally known at Berkeley as 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

44 teaching assistants (TAs), formally known at Berkeley as 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

44 teaching assistants (TAs), formally known at Berkeley as 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

44 teaching assistants (TAs), formally known at Berkeley as 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

1,300+ 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
An Introduction to Computer Science
What is Computer Science?
What is Computer Science?

The study of
What is Computer Science?

The study of computation, 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

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

Systems
What is Computer Science?

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

The study of 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?

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

The study of

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?

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

The study of...

- 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     Decision Making
Graphics                  Robotics
Security                  Natural Language Processing
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

Artificial Intelligence

- Graphics
- Robotics

Graphics

- Security
- Natural Language Processing

Security

- Networking

Networking

- Programming Languages

Programming Languages

- Theory

Theory

- Scientific Computing

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

Answering Questions
Decision Making
Robotics
Natural Language Processing

...
What is Computer Science?

The study of how to solve those problems, and what techniques lead to effective solutions.

What problems can be solved using computation?

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

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 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
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

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

Spring 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

Spring 2018: Dan Garcia

20+ person waitlist
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

Spring 2018: Dan Garcia

20+ person waitlist

More info: http://cs10.org/sp18/
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

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

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

Spring 2018: Ani Adhikari
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

Spring 2018: Ani Adhikari

100+ 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

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

Spring 2018: Ani Adhikari

100+ person waitlist

More info: http://data8.org/sp18
Course Policies
Course Policies
Course Policies

Learning
Course Policies

Learning Community
Course Policies

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

\[ 18 + 69 = 87 \]

\[ \frac{6}{23} \]
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

\[ 18 + 69 \]
\[ \frac{6}{23} \]
\[ \sin \pi \]
\[ \sqrt{3493161} \]
\[ \sum_{i=1}^{100} i \]
\[ | -1869| \]
\[ \begin{pmatrix} 69 \\ 18 \end{pmatrix} \]
Types of expressions

An expression describes a computation and evaluates to a value

\[
\begin{align*}
18 + 69 & \\
6 & \\
23 & \\
\sin \pi & \\
f(x) & \\
\sum_{i=1}^{100} i & \\
| - 1869| & \\
\sqrt{3493161} & \\
\end{align*}
\]
Types of expressions

An expression describes a computation and evaluates to a value

\[ 18 + 69 \]
\[ \frac{6}{23} \]
\[ \sin \pi \]
\[ 2^{100} \]
\[ f(x) \]
\[ \sum_{i=1}^{100} i \]
\[ | - 1869| \]
\[ \sqrt{3493161} \]
\[ (69) \]
\[ (18) \]
Types of expressions

An expression describes a computation and evaluates to a value

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

\[ 2^{100} \]

\[ f(x) \]

\[ \sum_{i=1}^{100} i \]

\[ | - 1869| \]

\[ \sqrt{3493161} \]

\[ (69) \]

\[ (18) \]
Types of expressions

An expression describes a computation and evaluates to a value

\[
\begin{align*}
18 + 69 & \quad \frac{6}{23} & \quad \sin \pi & \quad \log_2 1024 \\
2^{100} & \quad f(x) & \quad \sqrt{3493161} & \\
7 \mod 2 & \quad \sum_{i=1}^{100} i & & \\
| -1869| & & (69) & (18)
\end{align*}
\]
Types of expressions

An expression describes a computation and evaluates to a value

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

\[ 2^{100} \]

\[ f(x) \]

\[ 7 \mod 2 \]

\[ | - 1869| \]

\[ \sum_{i=1}^{100} i \]

\[ \sqrt{3493161} \]

\[ \lim_{x \to \infty} \frac{1}{x} \]

\[ \binom{69}{18} \]
Types of expressions

An expression describes a computation and evaluates to a value

\[ 18 + 69 \]
\[ \frac{6}{23} \]
\[ \sin \pi \]
\[ \log_2 1024 \]
\[ 2^{100} \]
\[ f(x) \]
\[ 7 \mod 2 \]
\[ \left| -1869 \right| \]
\[ \sum_{i=1}^{100} i \]
\[ \sqrt{3493161} \]
\[ \lim_{x \to \infty} \frac{1}{x} \]
\[ \binom{69}{18} \]
Call Expressions in Python

All expressions can use function call notation

(Demo)
Anatomy of a Call Expression
Anatomy of a Call Expression

\[
\text{add} \quad (\quad 2 \quad , \quad 3 \quad )
\]
Anatomy of a Call Expression

\[ \text{add} \ ( \ 2 \ , \ 3 \ ) \]
Anatomy of a Call Expression

\[\text{add} \quad ( \quad 2 \quad , \quad 3 \quad )\]

Operator
Anatomy of a Call Expression

\[
\begin{array}{c}
\text{add} \\
\hline
\text{Operator} \\
\hline
\text{Operand} \\
\hline
2 \\
\hline
, \\
\hline
3 \\
\hline
\text{Operand}
\end{array}
\]
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

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

\[
\text{mul}(\text{add}(4, \text{mul}(4, 6)), \text{add}(3, 5))
\]
Evaluating Nested Expressions

\[ \text{mul}(\text{add}(4, \text{mul}(4, 6)), \text{add}(3, 5)) \]
Evaluating Nested Expressions

mul(add(4, mul(4, 6)), add(3, 5))
Evaluating Nested Expressions

\[
\text{mul}\left(\text{add}(4, \text{mul}(4, 6)), \text{add}(3, 5)\right)
\]
Evaluating Nested Expressions

\[ \text{mul}\left(\text{add}(4, \text{mul}(4, 6)), \text{add}(3, 5)\right) \]
Evaluating Nested Expressions

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

mul

add(4, mul(4, 6))

add

4
Evaluating Nested Expressions

\[
mul\left(add(4, mul(4, 6)), add(3, 5)\right)
\]
Evaluating Nested Expressions

\( \text{mul} (\text{add}(4, \text{mul}(4, 6)), \text{add}(3, 5)) \)
Evaluating Nested Expressions

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

mul

add(4, mul(4, 6))

add

4

mul(4, 6)

mul

4

6
Evaluating Nested Expressions

\[ \text{mul}(\text{add}(4, \text{mul}(4, 6)), \text{add}(3, 5)) \]
Evaluating Nested Expressions

\[ \text{mul}\left(\text{add}(4, \text{mul}(4, 6)), \text{add}(3, 5)\right) \]

\[ \text{mul}\left(\text{add}(4, 24), 5\right) \]

\[ 20 \]
Evaluating Nested Expressions

\[
\text{mul}(\text{add}(4, \text{mul}(4, 6)), \text{add}(3, 5))
\]
Evaluating Nested Expressions

\[
\text{mul}(\text{add}(4, \text{mul}(4, 6)), \text{add}(3, 5))
\]

\[
\text{mul}(\text{add}(4, 24), \text{add}(3, 5))
\]

\[
\text{mul}(\text{add}(4, \text{mul}(4, 6)), \text{add}(3, 5))
\]

\[
\text{mul}(\text{add}(4, 24), \text{add}(3, 5))
\]
Evaluating Nested Expressions

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

```
mul(add(4, mul(4, 6)), add(3, 5))
mul
28
add(4, mul(4, 6))
add
4
24
mul(4, 6)
mul
4
6
mul
add
3
5
```
Evaluating Nested Expressions

\[
\text{mul}(\text{add}(4, \text{mul}(4, 6)), \text{add}(3, 5))
\]
Evaluating Nested Expressions

\[\text{mul} \left( \text{add}(4, \text{mul}(4, 6)), \text{add}(3, 5) \right)\]
Evaluating Nested Expressions

\[
\text{mul}(\text{add}(4, \text{mul}(4, 6)), \text{add}(3, 5))
\]
Evaluating Nested Expressions

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

Evaluation:

```
mul
  add
    4
      mul
        4
          6
    24
  28
add
  3
  5
```
Evaluating Nested Expressions

\[
\text{mul}(\text{add}(4, \text{mul}(4, 6)), \text{add}(3, 5))
\]
Evaluating Nested Expressions

Expression tree
Evaluating Nested Expressions

Expression tree

Operand subexpression

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

mul(24)

add(28)

mul(4, 6)

mul(4, 6)

add(4)

add(3, 5)

mul(4, 6)

mul(4, 6)

add(3)

add(5)

Expression tree
Evaluating Nested Expressions

Expression tree

Operand subexpression

Value of subexpression

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

mul

add(4, mul(4, 6))

add(3, 5)

mul

add

add

mul

4

28

24

8

3

5

4

6

224
Evaluating Nested Expressions

Expression tree

Operand subexpression

Value of subexpression

1st argument to mul

mul(\text{add}(4, \text{mul}(4, 6)), \text{add}(3, 5))

mul(\text{add}(4, \text{mul}(4, 6)))

\text{add}(4, \text{mul}(4, 6))

mul(4, 6)

\text{mul}(4, 6)

\text{add}(3, 5)

\text{add}(3, 5)

4

24

8

add

4

6

mul

add

3
Evaluating Nested Expressions

\[ \text{mul}(\text{add}(4, \text{mul}(4, 6)), \text{add}(3, 5)) \]

Expression tree:

Operand subexpression: \( \text{mul}(\text{add}(4, \text{mul}(4, 6)), \text{add}(3, 5)) \)

Value of the whole expression: 224

Value of subexpression: \( \text{mul}(28, 8) \)

1st argument to \( \text{mul} \): 28

Operand subexpression: \( \text{add}(4, \text{mul}(4, 6)) \)

Value of subexpression: 24

1st argument to \( \text{add} \): 4

Operand subexpression: \( \text{mul}(4, 6) \)

Value of subexpression: 24

1st argument to \( \text{mul} \): 4

Operand subexpression: \( \text{add}(3, 5) \)

Value of subexpression: 8

1st argument to \( \text{add} \): 3

Operand subexpression: \( \text{add}(4, \text{mul}(4, 6)) \)

Value of subexpression: 28

1st argument to \( \text{add} \): 4

Operand subexpression: \( \text{mul}(4, 6) \)

Value of subexpression: 24

1st argument to \( \text{mul} \): 4

Operand subexpression: \( \text{add}(3, 5) \)

Value of subexpression: 8

1st argument to \( \text{add} \): 3
Functions, Values, Objects, Interpreters, and Data

(Demo)