Welcome to CS 61A!

I'm John DeNero
Welcome to CS 61A!

I'm John DeNero
Welcome to CS 61A!

I'm John DeNero
Welcome to CS 61A!

I'm John DeNero
Welcome to CS 61A!

I'm John DeNero

How to contact John: denero@berkeley.edu
Welcome to CS 61A!

I'm John DeNero

How to contact John:
denero@berkeley.edu
piazza.com/berkeley/fall2016/cs61a
Welcome to CS 61A!

I'm John DeNero

How to contact John:  
denero@berkeley.edu  
piazza.com/berkeley/fall2016/cs61a

John's office hours:  
781 Soda  
Monday & Wednesday 11am – 12pm  
By appointment: denero.org/meet
The 61A Community
The 61A Community

45 undergraduate student instructors / teaching assistants (TAs):
The 61A Community

45 undergraduate student instructors / teaching assistants (TAs):
• Teach lab & discussion sections
The 61A Community

45 undergraduate student instructors / teaching assistants (TAs):
• Teach lab & discussion sections
• Hold office hours
The 61A Community

45 undergraduate student instructors / teaching assistants (TAs):
• Teach lab & discussion sections
• Hold office hours
• Lots of other stuff: develop assignments, grade exams, etc.
The 61A Community

45 undergraduate student instructors / teaching assistants (TAs):
• Teach lab & discussion sections
• Hold office hours
• Lots of other stuff: develop assignments, grade exams, etc.

45+ tutors & mentors:
The 61A Community

45 undergraduate student instructors / teaching assistants (TAs):
• Teach lab & discussion sections
• Hold office hours
• Lots of other stuff: develop assignments, grade exams, etc.

45+ tutors & mentors:
• Teach mentoring sections
The 61A Community

45 undergraduate student instructors / teaching assistants (TAs):
• Teach lab & discussion sections
• Hold office hours
• Lots of other stuff: develop assignments, grade exams, etc.

45+ tutors & mentors:
• Teach mentoring sections
• Hold office hours
The 61A Community

45 undergraduate student instructors / teaching assistants (TAs):
• Teach lab & discussion sections
• Hold office hours
• Lots of other stuff: develop assignments, grade exams, etc.

45+ tutors & mentors:
• Teach mentoring sections
• Hold office hours
• Lots of other stuff: homework parties, mastery sections, etc.
The 61A Community

45 undergraduate student instructors / teaching assistants (TAs):
• Teach lab & discussion sections
• Hold office hours
• Lots of other stuff: develop assignments, grade exams, etc.

45+ tutors & mentors:
• Teach mentoring sections
• Hold office hours
• Lots of other stuff: homework parties, mastery sections, etc.

200+ lab assistants help answer your individual questions
The 61A Community

45 undergraduate student instructors / teaching assistants (TAs):
- Teach lab & discussion sections
- Hold office hours
- Lots of other stuff: develop assignments, grade exams, etc.

45+ tutors & mentors:
- Teach mentoring sections
- Hold office hours
- Lots of other stuff: homework parties, mastery sections, etc.

200+ lab assistants help answer your individual questions

1,500+ 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](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 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?

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?

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
Graphics
Security
Networking
Programming Languages
Theory
Scientific Computing

Decision Making
Robotics
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
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
Answering Questions
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
Robotics
Decision Making

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.

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

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
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

Taught in Fall 2016 by 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

Taught in Fall 2016 by Dan Garcia

More info: cs10.org
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

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

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

Great programming practice for CS 61A

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

Taught in Fall 2016 by Ani Adhikari
Data Science 8: Foundations of Data Science

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

Great programming practice for CS 61A

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

Taught in Fall 2016 by Ani Adhikari

More info: data8.org & databears.berkeley.edu
Course Policies
Course Policies
Learning
Course Policies

Learning

Community
Learning

Community

Course Staff
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!
• Homework can be completed with a partner
Collaboration

**Asking questions is highly encouraged**

- Discuss everything with each other; learn from your fellow students!
- Homework can be completed with a partner
- Projects should be completed with a partner
Collaboration

Asking questions is highly encouraged

• Discuss everything with each other; learn from your fellow students!
• Homework can be completed with a partner
• Projects should 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!
• Homework can be completed with a partner
• Projects should 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!
• Homework can be completed with a partner
• Projects should 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 partner
Collaboration

Asking questions is highly encouraged

• Discuss everything with each other; learn from your fellow students!
• Homework can be completed with a partner
• Projects should 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 partner
• Copying project solutions causes people to fail
Collaboration

Asking questions is highly encouraged
• Discuss everything with each other; learn from your fellow students!
• Homework can be completed with a partner
• Projects should 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 partner
• Copying project solutions causes people to fail
• 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!
• Homework can be completed with a partner
• Projects should 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 partner
• Copying project solutions causes people to fail
• 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!
• Homework can be completed with a partner
• Projects should 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 partner
• Copying project solutions causes people to fail
• 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!
- Homework can be completed with a partner
- Projects should 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 partner
- Copying project solutions causes people to fail
- 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

\[ 18 + 69 = 87 \]
Types of expressions

An expression describes a computation and evaluates to a value

$$18 + 69$$

$$\sqrt{3493161}$$
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

\[
18 + 69 \quad \frac{6}{23} \quad \sin \pi \quad \sqrt{3493161} \quad | - 1869|
\]
Types of expressions

An expression describes a computation and evaluates to a value

\[
18 + 69 \quad \frac{6}{23} \quad \sin \pi \quad \sqrt{3493161} \quad \sum_{i=1}^{100} i \quad | - 1869 |
\]
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

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

An expression describes a computation and evaluates to a value

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

\[ 2^{100} \]

\[ f(x) \]

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

\[ | - 1869| \]

\[ \sqrt{3493161} \]

\( (69) \) (18)
Types of expressions

An expression describes a computation and evaluates to a value

\[ 18 + 69 \quad 6 \quad \sin \pi \quad \log_2 1024 \]

\[ \frac{2^{100}}{23} \]

\[ f(x) \]

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

\[ | -1869| \]

\[ \sqrt{3493161} \]

\[ (69) \quad (18) \]
Types of expressions

An expression describes a computation and evaluates to a value

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

An expression describes a computation and evaluates to a value

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

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

\[
7 \mod 2 \quad \sum_{i=1}^{100} i \quad \lim_{x \to \infty} \frac{1}{x}
\]

\[
| -1869| \quad \binom{69}{18}
\]
Types of expressions

An expression describes a computation and evaluates to a value

\[ 18 + 69 \]

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

\[ \sin \pi \]

\[ \log_2 1024 \]

\[ 2^{100} \]

\[ f(x) \]

\[ 7 \mod 2 \]

\[ | -1869| \]

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

\[ \sqrt{3493161} \]

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

\[ \binom{69}{18} \]
Call Expressions in Python

All expressions can use function call notation
(Demo)
Anatomy of a Call Expression
Anatomy of a Call Expression

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

```
add ( 2 , 3 )
```
Anatomy of a Call Expression

```
add ( 2, 3 )
```

Operator
Anatomy of a Call Expression

add ( 2 , 3 )

Operator
Operand
Operand
Anatomy of a Call Expression

```
add ( 2 , 3 )

Operator     Operand       Operand
```

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:

```
add ( 2 , 3 )
```

Operator   Operand   Operand
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

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

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

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

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

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

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

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

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

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

```
mul
```

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

```
add
```

```
4
```

```
24
```

```
mul(4, 6)
```

```
mul
```

```
4
```

```
6
```
Evaluating Nested Expressions

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

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

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

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

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

Evaluation:

- \[\text{mul}(\text{add}(4, \text{mul}(4, 6)), \text{add}(3, 5))\]
  \[= \text{mul}(28, \text{add}(3, 5))\]
  \[= \text{mul}(28, 8)\]
  \[= 224\]
Evaluating Nested Expressions

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

Diagram:
- **mul**
  - **28**
    - **add**
      - **4**
      - **mul**(4, 6)
    - **mul**(4, 6)
  - **add**(3, 5)
  - **add**
    - **3**
    - **5**
Evaluating Nested Expressions

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

- `mul(4, 6)` evaluates to 24
- `add(3, 5)` evaluates to 8
- `mul(add(4, mul(4, 6)), add(3, 5))` evaluates to 8 * 24 = 192
Evaluating Nested Expressions

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

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

Graph representation:

- Node 224: \( \text{mul}(\text{add}(4, \text{mul}(4, 6)), \text{add}(3, 5)) \)
- Node 28: \( \text{add}(4, \text{mul}(4, 6)) \)
- Node 24: \( \text{mul}(4, 6) \)
- Node 8: \( \text{add}(3, 5) \)
- Node 4: \( \text{add}(3, 5) \)
- Node 6: \( \text{add}(3, 5) \)

Operations:
- \( \text{mul} \)
- \( \text{add} \)
Evaluating Nested Expressions

Expression tree
Evaluating Nested Expressions

Expression tree

Operand subexpression

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

mul

add(4, mul(4, 6))

add

mul(4, 6)

mul

mul(4, 6)

add(3, 5)

add

3

5

24

28

224
Evaluating Nested Expressions
Evaluating Nested Expressions

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

Operand subexpression

Value of subexpression

1st argument to `mul`

Expression tree
Evaluating Nested Expressions

Expression tree

Operand subexpression
Value of subexpression
Value of the whole expression

1st argument to mul

Expression tree
Functions, Objects, and Interpreters

(Demo)