Welcome to CS 61A!

Lecture will begin at 2:10pm.

(Until then, you just get to watch us type.)

Looking for something to do?
Try reading the "Syllabus" link at the top of cs61a.org

Questions?
There is an Ed thread for this lecture

Welcome to CS 61A

Your Instructors

John DeNero
denero@berkeley.edu

Hany Farid
hfarid@berkeley.edu

Your Instructors

John DeNero
denero@berkeley.edu

CS 61A instructor many times

Hany Farid
hfarid@berkeley.edu

Your Instructors

John DeNero
denero@berkeley.edu

CS 61A instructor many times

Teaching Professor in EECS and help with the Data Science undergraduate program

Hany Farid
hfarid@berkeley.edu

Your Instructors

John DeNero
denero@berkeley.edu

CS 61A instructor many times

Teaching Professor in EECS and help with the Data Science undergraduate program

Research focused on human-computer interaction with text generation AI

Hany Farid
hfarid@berkeley.edu

Your Instructors

John DeNero
denero@berkeley.edu

CS 61A instructor many times

Teaching Professor in EECS and help with the Data Science undergraduate program

Research focused on human-computer interaction with text generation AI

Hany Farid
hfarid@berkeley.edu

Office hours start next week (Jan 24):

Your Instructors

John DeNero
denero@berkeley.edu

CS 61A instructor many times

Teaching Professor in EECS and help with the Data Science undergraduate program

Research focused on human-computer interaction with text generation AI

Hany Farid
hfarid@berkeley.edu

Office hours start next week (Jan 24):

- 2-3:30 Tuesdays in 101B Warren Hall

Your Instructors

John DeNero
denero@berkeley.edu

CS 61A instructor many times

Teaching Professor in EECS and help with the Data Science undergraduate program

Research focused on human-computer interaction with text generation AI

Hany Farid
hfarid@berkeley.edu

Intro CS instructor many times

Office hours start next week (Jan 24):

- 2-3:30 Tuesdays in 101B Warren Hall

Your Instructors

John DeNero
denero@berkeley.edu

CS 61A instructor many times

Teaching Professor in EECS and help with the Data Science undergraduate program

Research focused on human-computer interaction with text generation AI

Hany Farid
hfarid@berkeley.edu

Intro CS instructor many times

Professor in both the School of Information and EECS

Office hours start next week (Jan 24):

- 2-3:30 Tuesdays in 101B Warren Hall

Your Instructors

John DeNero
denero@berkeley.edu

CS 61A instructor many times

Teaching Professor in EECS and help with the Data Science undergraduate program

Research focused on human-computer interaction with text generation AI

Hany Farid
hfarid@berkeley.edu

Intro CS instructor many times

Professor in both the School of Information and EECS

Research focused on digital forensics, forensic science, misinformation, image analysis, and human perception

Office hours start next week (Jan 24):

- 2-3:30 Tuesdays in 101B Warren Hall

Your Instructors

John DeNero
denero@berkeley.edu

CS 61A instructor many times

Teaching Professor in EECS and help with the Data Science undergraduate program

Research focused on human-computer interaction with text generation AI

Office hours start next week (Jan 24):

- 2-3:30 Tuesdays in 101B Warren Hall

Hany Farid
hfarid@berkeley.edu

Intro CS instructor many times

Professor in both the School of Information and EECS

Research focused on digital forensics, forensic science, misinformation, image analysis, and human perception

Office hours:

Your Instructors

John DeNero
denero@berkeley.edu

CS 61A instructor many times

Teaching Professor in EECS and help with the Data Science undergraduate program

Research focused on human-computer interaction with text generation AI

Office hours start next week (Jan 24):

- 2-3:30 Tuesdays in 101B Warren Hall

Hany Farid
hfarid@berkeley.edu

Intro CS instructor many times

Professor in both the School of Information and EECS

Research focused on digital forensics, forensic science, misinformation, image analysis, and human perception

Office hours:

- TBD

61A Course Staff

https://cs61a.org/TAs/ https://cs61a.org/tutors/ (coming soon)

About the Course

Parts of the Course

Parts of the Course

Lecture: Videos/recordings posted to cs61a.org

Parts of the Course

Lecture: Videos/recordings posted to cs61a.org

Lab: Practice ideas from lecture on a computer (often in pairs)

Parts of the Course

Lecture: Videos/recordings posted to cs61a.org

Lab: Practice ideas from lecture on a computer (often in pairs)

Discussion: Practice ideas from lecture on paper (often in groups)

Parts of the Course

Lecture: Videos/recordings posted to cs61a.org

Lab: Practice ideas from lecture on a computer (often in pairs)

Discussion: Practice ideas from lecture on paper (often in groups)

Assignments: Weekly homework + 4 large projects

Parts of the Course

Lecture: Videos/recordings posted to cs61a.org

Lab: Practice ideas from lecture on a computer (often in pairs)

Discussion: Practice ideas from lecture on paper (often in groups)

Assignments: Weekly homework + 4 large projects

Staff Office Hours: Get 1-on-1 help with assignments & work with peers

Parts of the Course

Lecture: Videos/recordings posted to cs61a.org

Lab: Practice ideas from lecture on a computer (often in pairs)

Discussion: Practice ideas from lecture on paper (often in groups)

Assignments: Weekly homework + 4 large projects

Staff Office Hours: Get 1-on-1 help with assignments & work with peers

Online textbook: http://composingprograms.com

Parts of the Course

Lecture: Videos/recordings posted to cs61a.org

Lab: Practice ideas from lecture on a computer (often in pairs)

Discussion: Practice ideas from lecture on paper (often in groups)

Assignments: Weekly homework + 4 large projects

Staff Office Hours: Get 1-on-1 help with assignments & work with peers

Online textbook: http://composingprograms.com

Monday-Wednesday: Attend lab and complete the lab assignment

Parts of the Course

Lecture: Videos/recordings posted to cs61a.org
Lab: Practice ideas from lecture on a computer (often in pairs)
Discussion: Practice ideas from lecture on paper (often in groups)
Assignments: Weekly homework + 4 large projects
Staff Office Hours: Get 1-on-1 help with assignments & work with peers
Online textbook: http://composingprograms.com

Monday-Wednesday: Attend lab and complete the lab assignment
Wednesday-Friday: Attend discussion

Parts of the Course

Lecture: Videos/recordings posted to cs61a.org
Lab: Practice ideas from lecture on a computer (often in pairs)
Discussion: Practice ideas from lecture on paper (often in groups)
Assignments: Weekly homework + 4 large projects
Staff Office Hours: Get 1-on-1 help with assignments & work with peers
Online textbook: http://composingprograms.com

Monday-Wednesday: Attend lab and complete the lab assignment
Wednesday-Friday: Attend discussion
Watch (or go to) lecture before you show up to lab/discussion!

Parts of the Course

Lecture: Videos/recordings posted to cs61a.org
Lab: Practice ideas from lecture on a computer (often in pairs)
Discussion: Practice ideas from lecture on paper (often in groups)
Assignments: Weekly homework + 4 large projects
Staff Office Hours: Get 1-on-1 help with assignments & work with peers
Online textbook: http://composingprograms.com

Monday-Wednesday: Attend lab and complete the lab assignment
Wednesday-Friday: Attend discussion
Watch (or go to) lecture before you show up to lab/discussion!
Sunday/Monday/Tuesday: Start on the homework or project

Parts of the Course

Lecture: Videos/recordings posted to cs61a.org
Lab: Practice ideas from lecture on a computer (often in pairs)
Discussion: Practice ideas from lecture on paper (often in groups)
Assignments: Weekly homework + 4 large projects
Staff Office Hours: Get 1-on-1 help with assignments & work with peers
Online textbook: http://composingprograms.com

Monday-Wednesday: Attend lab and complete the lab assignment
Wednesday-Friday: Attend discussion
Watch (or go to) lecture before you show up to lab/discussion!
Sunday/Monday/Tuesday: Start on the homework or project
Wednesday/Thursday: Finish the homework or project

Parts of the Course

Lecture: Videos/recordings posted to cs61a.org
Lab: Practice ideas from lecture on a computer (often in pairs)
Discussion: Practice ideas from lecture on paper (often in groups)
Assignments: Weekly homework + 4 large projects
Staff Office Hours: Get 1-on-1 help with assignments & work with peers
Online textbook: http://composingprograms.com

Monday-Wednesday: Attend lab and complete the lab assignment
Wednesday-Friday: Attend discussion
Watch (or go to) lecture before you show up to lab/discussion!
Sunday/Monday/Tuesday: Start on the homework or project
Wednesday/Thursday: Finish the homework or project
Friday: Finish projects you didn't finish on Thursday

Asking Questions

?

Asking Questions

?

Ed: All staff (private posts) and students (public posts)

Asking Questions

?

Ed: All staff (private posts) and students (public posts)

cs61a@berkeley.edu: Head TAs and instructors

Asking Questions

?

Ed: All staff (private posts) and students (public posts)

cs61a@berkeley.edu: Head TAs and instructors

denero@berkeley.edu or **hfarid@berkeley.edu:** Might be slow

Asking Questions

?

Ed: All staff (private posts) and students (public posts)

cs61a@berkeley.edu: Head TAs and instructors

denero@berkeley.edu or **hfarid@berkeley.edu:** Might be slow

cs61a.org: Self-service answers to many questions

Asking Questions

?

Ed: All staff (private posts) and students (public posts)

cs61a@berkeley.edu: Head TAs and instructors

denero@berkeley.edu or **hfarid@berkeley.edu:** Might be slow

cs61a.org: Self-service answers to many questions

cs61a.org/contact/: Even more ways to reach the course staff

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

A course about managing complexity

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

A course about managing complexity

Mastering abstraction

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

A course about managing complexity

Mastering abstraction

An introduction to programming

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

A course about managing complexity

- Mastering abstraction

An introduction to programming

- Full understanding of Python fundamentals

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

A course about managing complexity

- Mastering abstraction

An introduction to programming

- Full understanding of Python fundamentals
- Combining multiple ideas in large projects

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

A course about managing complexity

- Mastering abstraction

An introduction to programming

- Full understanding of Python fundamentals
- Combining multiple ideas in large projects
- How computers interpret programming languages

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

A course about managing complexity

- Mastering abstraction

An introduction to programming

- Full understanding of Python fundamentals
- Combining multiple ideas in large projects
- How computers interpret programming languages
- Different kinds of programming languages

Should you take CS 61A?

According to the Syllabus

According to the Syllabus

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

According to the Syllabus

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

- Taking the course without any prior programming experience is typically very challenging.

According to the Syllabus

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

- Taking the course without any prior programming experience is typically very challenging.
- Most CS 61A students have had significant prior programming experience.

According to the Syllabus

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

- Taking the course without any prior programming experience is typically very challenging.
- Most CS 61A students have had significant prior programming experience.
- Students who take the course without prior programming experience typically must work substantially harder to master the material and tend to receive lower final grades in the course.

According to the Syllabus

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

- Taking the course without any prior programming experience is typically very challenging.
- Most CS 61A students have had significant prior programming experience.
- Students who take the course without prior programming experience typically must work substantially harder to master the material and tend to receive lower final grades in the course.

Students who take the course later often get more out of it due to increased understanding.

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

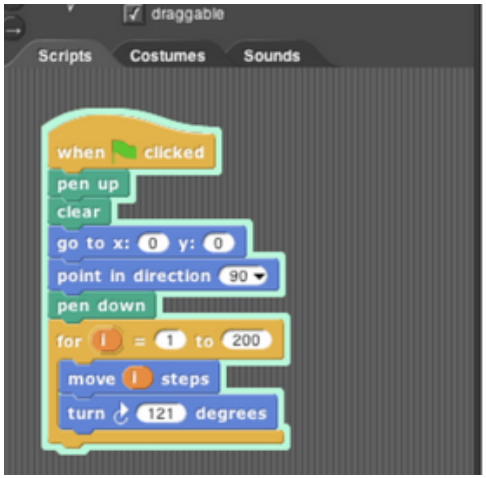
Designed for students without prior experience

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

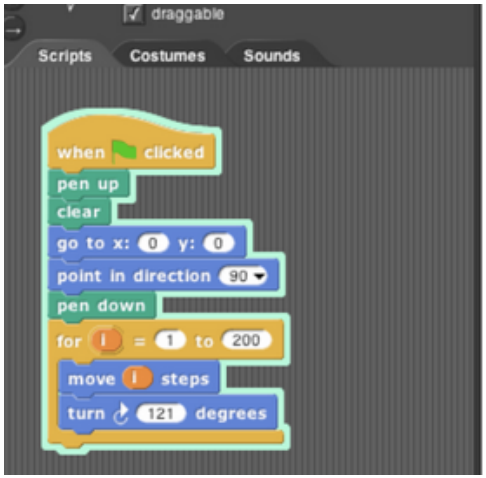
A programming environment created by Berkeley, now used in courses around the world and online

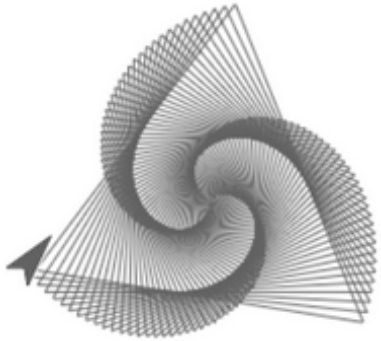


CS 10: The Beauty and Joy of Computing

Designed for students without prior experience

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



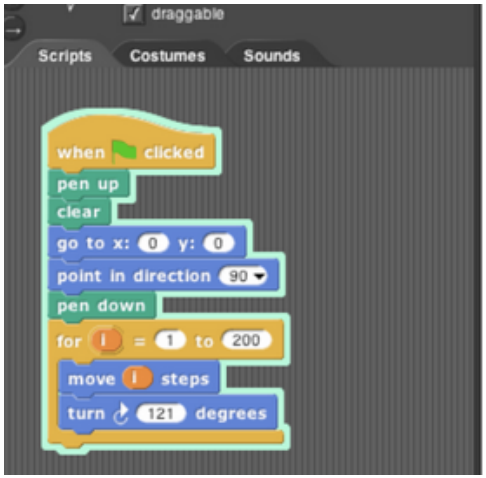


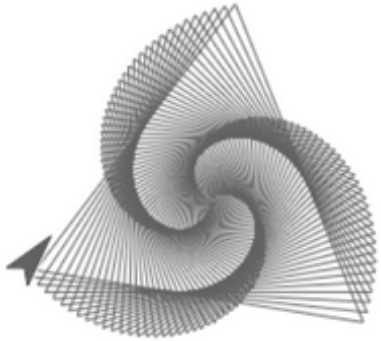
CS 10: The Beauty and Joy of Computing

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





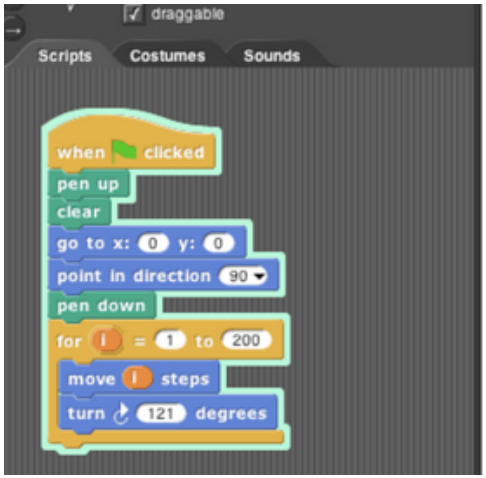
CS 10: The Beauty and Joy of Computing

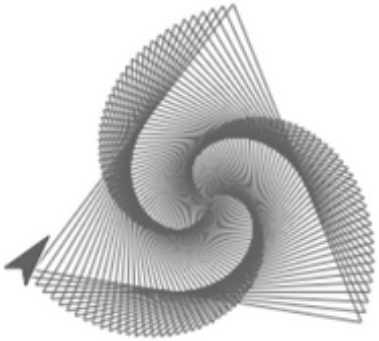
Designed for students without prior experience

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

If you might switch to CS 10, start attending ASAP and enroll soon before it fills!





CS 10: The Beauty and Joy of Computing

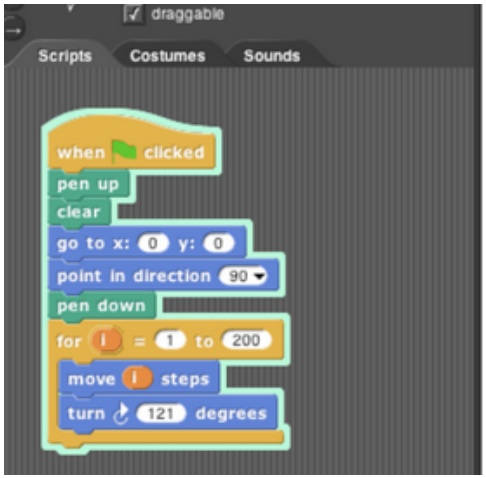
Designed for students without prior experience

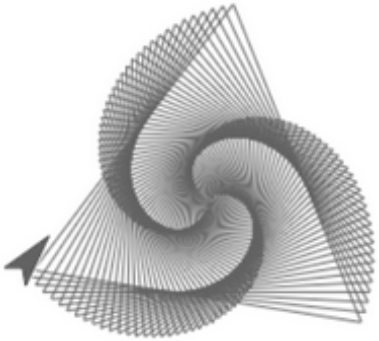
A programming environment created by Berkeley, now used in courses around the world and online

An introduction to fundamentals (& Python) that sets students up for success in CS 61A

If you might switch to CS 10, start attending ASAP and enroll soon before it fills!

More info: http://cs10.org/





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

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

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

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

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

- Two programming projects (instead of four) that are adapted from CS 61A projects

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

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

- Two programming projects (instead of four) that are adapted from CS 61A projects
- Everything you need to know to continue on to CS 61B

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

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

- Two programming projects (instead of four) that are adapted from CS 61A projects
- Everything you need to know to continue on to CS 61B
- Omits the unit on how programs run other programs

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

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

- Two programming projects (instead of four) that are adapted from CS 61A projects
- Everything you need to know to continue on to CS 61B
- Omits the unit on how programs run other programs

For students taking Data 8 (Foundations of Data Science) or who took it already

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

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

- Two programming projects (instead of four) that are adapted from CS 61A projects
- Everything you need to know to continue on to CS 61B
- Omits the unit on how programs run other programs

For students taking Data 8 (Foundations of Data Science) or who took it already

We're investigating expansion options

Course Policies

Course Policies

Learning

Learning

Community

Learning

Community

Course Staff

Course Policies

Learning

Community

Course Staff

Details...

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

Getting Help

If you're struggling, let us know.

If you need more time, ask for it.

If you need special accommodations, make an appointment.

Collaboration

Collaboration

Working together is highly encouraged

Collaboration

Working together is highly encouraged

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

Collaboration

Working together is highly encouraged

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

Collaboration

Working together 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

Working together 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

What constitutes academic misconduct?

Collaboration

Working together 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

What constitutes academic misconduct?

- *Please* don’t look at someone else's code!
  Exceptions: lab, your project partner, or **after you already solved the problem**

Collaboration

Working together 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

What constitutes academic misconduct?

- *Please* don’t look at someone else's code!
  Exceptions: lab, your project partner, or **after you already solved the problem**
- *Please* don't tell other people the answers! You can point them to what is wrong and describe how to fix it or show them a related example

Collaboration

Working together 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

What constitutes academic misconduct?

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

Collaboration

Working together 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

What constitutes academic misconduct?

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

(Switch)

75% of students report cheating on at least one exam per semester

(~20% are probably lying)

75% of students report cheating on at least one exam per semester
(~20% are probably lying)

50% of faculty ignore cheating in their class
(not us!)

75% of students report cheating on at least one exam per semester
(~20% are probably lying)

50% of faculty ignore cheating in their class
(not us!)

every semester dozens of students fail this course because of cheating

cheating is disrespectful

cheating is disrespectful

cheating destroys trust

cheating is disrespectful

cheating destroys trust

cheating is a trap

cheating is disrespectful

cheating destroys trust

cheating is a trap

cheating completely misses the point

it is just wrong

computer science is (much) more than coding


SIENNA LE


SIENNA
LE


SIENNA
LE


SIENNA


SIENNA

```python
# load some libraries
import cv2
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.image as image
from PIL import Image
from numpy import linalg as la
```

```python
# load and display image
im = image.imread('license.png')
plt.imshow(im)
plt.axis('off')
plt.show()
```

abstraction

```python
# specify world coordinates of rectangular license plate
X = np.array([100, 400, 400, 100])
Y = np.array([100, 100, 200, 200])

# specify image coordinates of distorted license plate
u = np.array([129.6, 169.5, 183.7, 142.9]);
v = np.array([216.5, 218.9, 86.4, 91.1]);

# display image with image coordinates
im = image.imread('license.png')
plt.scatter( u, v, s=100, facecolors='none', edgecolors='y')
plt.imshow(im)
plt.axis('off')
plt.show()
```

data structures

data visualization

function + form

```python
# estimate homography
A = np.zeros((8,9))
for i in range(0,4):
    A[2*i,:]   = [0, 0, 0, -X[i], -Y[i], -1, v[i]*X[i], v[i]*Y[i], v[i] ]
    A[2*i+1,:] = [X[i], Y[i], 1, 0, 0, 0, -u[i]*X[i], -u[i]*Y[i], -u[i] ]

At   = A.transpose()
L,V  = la.eig( At@A )
h    = V[:,-1] # minimial eigenvalue eigenvector (assumes that eigenvalues
H    = np.reshape(h,(3,3))
H    = la.inv(H)

# warp source image based on homography
im_warp = cv2.warpPerspective(im, H, (2*im.shape[1],im.shape[0]))

# display rectified image
plt.imshow(im_warp)
plt.gca().invert_yaxis()
plt.axis('off')
plt.show()
```

math

numerical methods

solving problems

ethical computing

Course Climate

Let's Stop Harassment & Discrimination

Let's Stop Harassment & Discrimination

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

Let's Stop Harassment & Discrimination

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

From the Berkeley Principles of Community:

Let's Stop Harassment & Discrimination

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

From the Berkeley Principles of Community:

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

Let's Stop Harassment & Discrimination

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

From the Berkeley Principles of Community:

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

From the EECS department mission:

Let's Stop Harassment & Discrimination

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

From the Berkeley Principles of Community:

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

From the EECS department mission:

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

Let's Stop Harassment & Discrimination

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

From the Berkeley Principles of Community:

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

From the EECS department mission:

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

All faculty and staff members are *mandated reporters*. If we ever receive a report of harassment, we must report to the Office for the Prevention of Harassment & Discrimination.

Let's Stop Harassment & Discrimination

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

From the Berkeley Principles of Community:

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

From the EECS department mission:

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

All faculty and staff members are *mandated reporters*. If we ever receive a report of harassment, we must report to the Office for the Prevention of Harassment & Discrimination.

- CS61A Anonymous feedback form: If you want to stay anonymous but make us aware of something happening in the course.

Let's Stop Harassment & Discrimination

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

From the Berkeley Principles of Community:

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

From the EECS department mission:

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

All faculty and staff members are *mandated reporters*. If we ever receive a report of harassment, we must report to the Office for the Prevention of Harassment & Discrimination.

- CS61A Anonymous feedback form: If you want to stay anonymous but make us aware of something happening in the course.
- EECS Student Climate & Incident Reporting Form: Informs the EECS department of any issues. You can also contact Susanne Kauer (skauer@berkeley.edu) directly.

The Best Approach to CS 61A

The Best Approach to CS 61A

Help each other understand concepts in the class, whether in section, on Ed, or in study groups, without expectation of anything in return.

The Best Approach to CS 61A

Help each other understand concepts in the class, whether in section, on Ed, or in study groups, without expectation of anything in return.

Be great project partners by listening to what your partner suggests and helping them understand the work you've done together.

The Best Approach to CS 61A

Help each other understand concepts in the class, whether in section, on Ed, or in study groups, without expectation of anything in return.

Be great project partners by listening to what your partner suggests and helping them understand the work you've done together.

Recognize that we're all valuable members of the CS community!

The Best Approach to CS 61A

Help each other understand concepts in the class, whether in section, on Ed, or in study groups, without expectation of anything in return.

Be great project partners by listening to what your partner suggests and helping them understand the work you've done together.

Recognize that we're all valuable members of the CS community!

Programming

(Demo)