# Welcome to CS106AP!

Programming Methodologies in Python

LELAND STANFORD JUNIOR UNIVERSITY
DIE LUFT DER FREIHEIT WEHT
1891

# Today’s questions

Why CS106A in Python?

Who are we?

What is CS106AP?

Who is Karel?

What’s next?

# Why CS106A in Python?

# Why **CS106A** in Python?

# Why learn computer science?

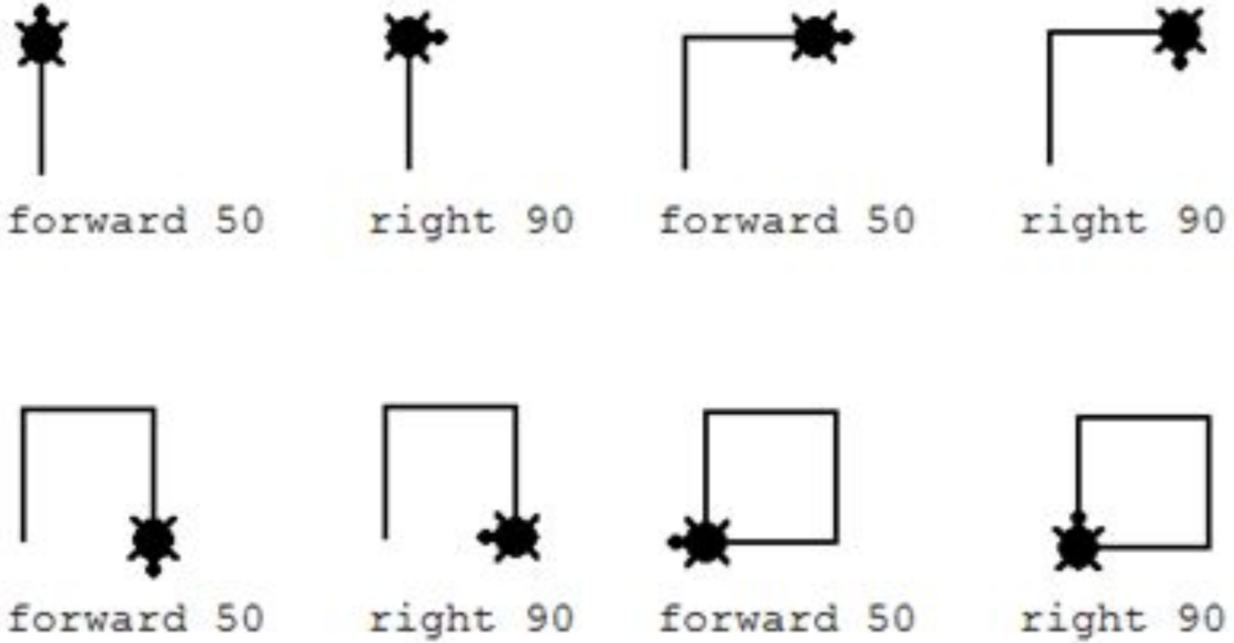
# Why learn computer science?

- Computer science is intimidating.
  - Jargon and “syntax”
  - Weird error messages
  - There are lots of people who say “I could never do computer science.”

# Why learn computer science?

- Computer science is intimidating.

- But the difficulty is superficial!



```
forward 50
right 90
forward 50
right 90
forward 50
right 90
forward 50
right 90
```

# Why learn computer science?

- Computer science is intimidating.

- But the difficulty is superficial!

- Code is all about creating.
    - Yes, it is a type of engineering.
    - But it’s also like learning another language…
    - Or mastering a new artform.

# Why learn computer science?

- Computer science is intimidating.
- But the difficulty is superficial!
- Code is all about creating.
- You gain an understanding of the computers and technology in your everyday life.

# Why CS106A in **Python**?

# Python as “programmer friendly”

# Python as “programmer friendly”

- More readable than most programming languages

# Python as “programmer friendly”

- More readable than most programming languages

- Popular in many different disciplines, especially for data processing

# Popular in many different disciplines!

- CEE 345: Network Analysis for Urban Systems
- COMM 177P: Programming in Journalism
- CS 375: Large-Scale Neural Network Modeling for Neuroscience
- ECON 288: Computational Economics
- ENGLISH 184D: Race, Gender, and Literary Digital Humanities
- HUMBIO 51: Big Data for Biologists - Decoding Genomic Function
- GENE 211: Genomics
- LINGUIST 278: Programming for Linguists
- MS&E 448: Big Financial Data and Algorithmic Trading
- PHYSICS 113: Computational Physics
- PSYCH 162: Brain Networks

*Slide courtesy of Sam Redmond, CS41*

# Python as “programmer friendly”

- More readable than most programming languages
- Popular in many different disciplines, especially for data processing
- Fast to write and run programs on any computer

# Python as “programmer friendly”

- More readable than most programming languages
- Popular in many different disciplines, especially for data processing
- Fast to write and run programs on any computer

Makes it easier for **anyone** to be a programmer!

# Python as “programmer friendly”

- More readable than most programming languages
- Popular in many different disciplines, especially for data processing
- Fast to write and run programs on any computer

Makes it easier for **anyone** to be a programmer!

YOU!

# Who are we?

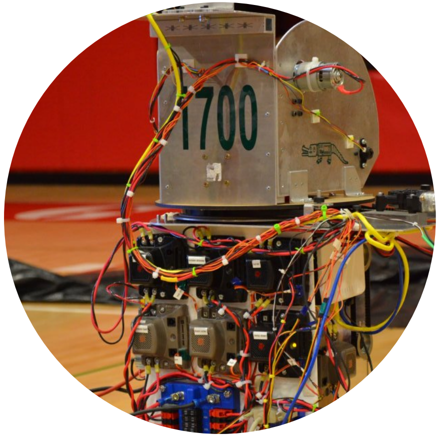
# Who am I?

# Kylie Jue

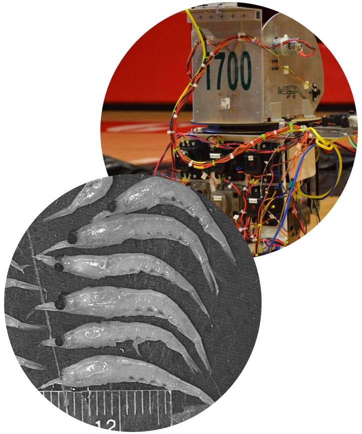
Kylie Jue

Kylie Jue

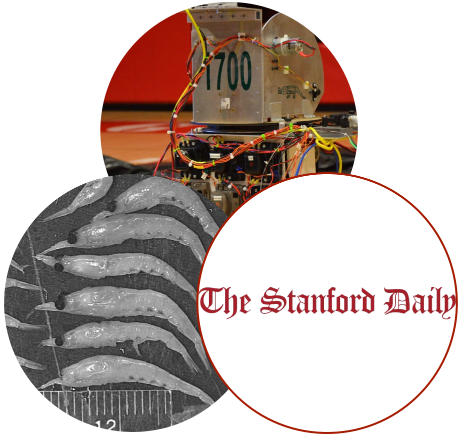
# Kylie Jue



# Kylie Jue

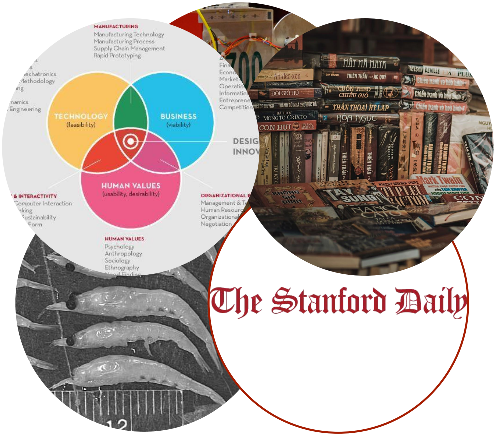


# Kylie Jue



# Kylie Jue

# Kylie Jue


MANUFACTURING
Manufacturing Technology
Manufacturing Process
Supply Chain Management
Rapid Prototyping
TECHNOLOGY
(feasibility)
BUSINESS
(viability)
HUMAN VALUES
(usability, desirability)
DESIG
INNOV
& INTERACTIVITY
Computer Interaction
Sustainability
Form
ORGANIZATIONAL
Management & T
Human Resour
Organizational
Negotiation
HUMAN VALUES
Psychology
Anthropology
Sociology
Ethnography
The Stanford Daily

Kylie Jue

# Who am I?

# Sonja Johnson-Yu

# Who am I?

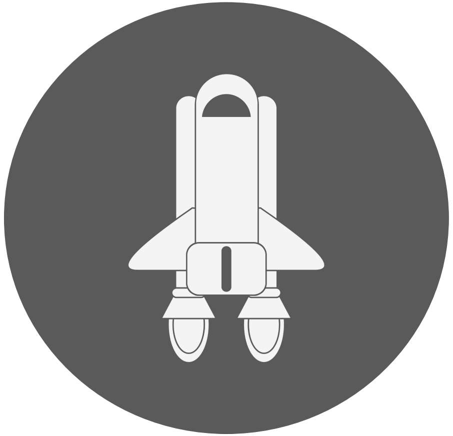
# Nick Bowman

Nick Bowman

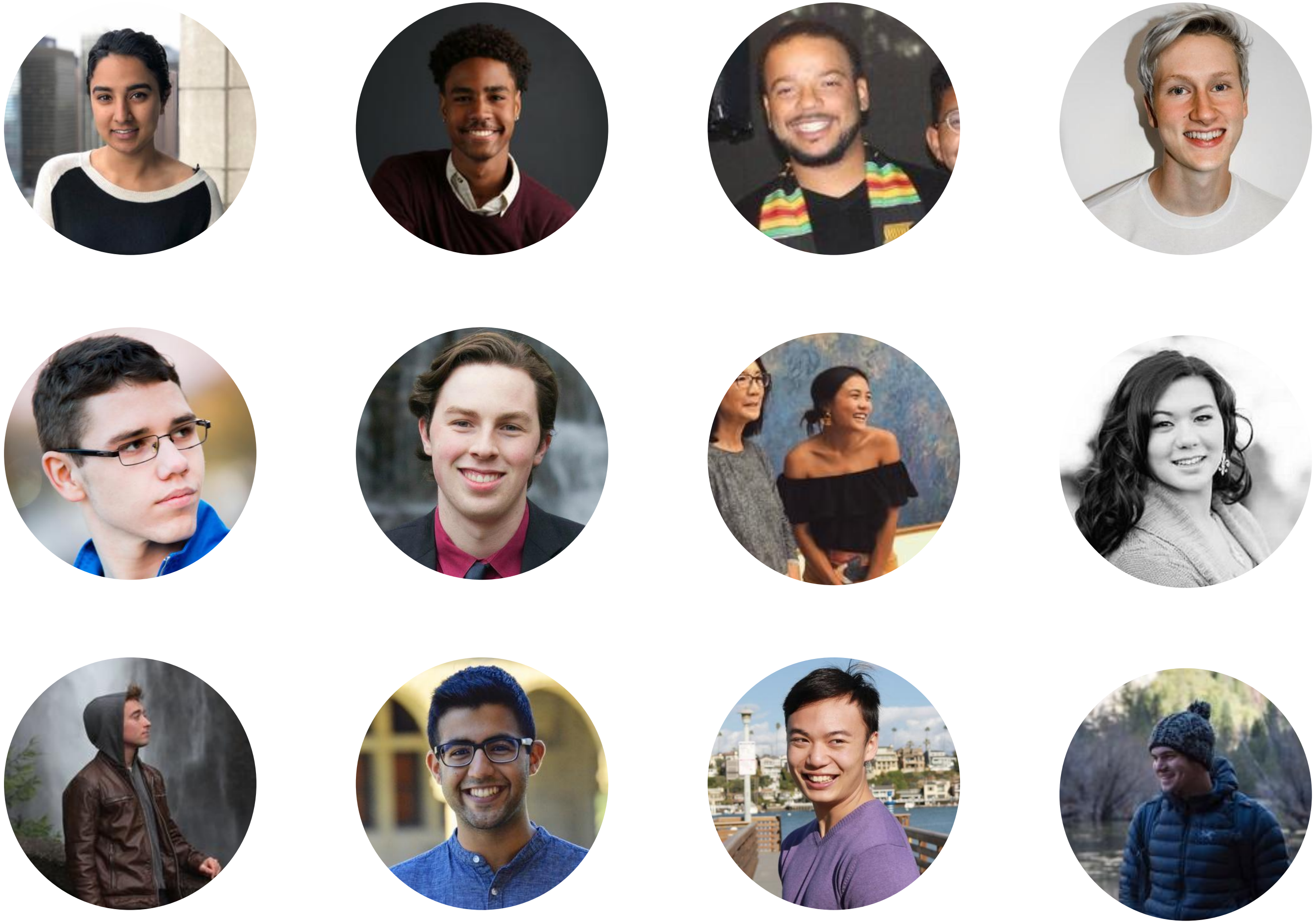
Nick Bowman

# Nick Bowman

# Nick Bowman



# Section Leaders



# What is CS106AP?

(Course topics)

# Learning goals

# Learning goals

- I am excited to use programming to solve real-world problems I encounter outside class, including those related to my major/career.

# Learning goals

- I am excited to use programming to solve real-world problems I encounter outside class, including those related to my major/career.

- I can break down complex problems into smaller subproblems by applying the logical reasoning skills I have gained from programming.

# Learning goals

- I am excited to use programming to solve real-world problems I encounter outside class, including those related to my major/career.
- I can break down complex problems into smaller subproblems by applying the logical reasoning skills I have gained from programming.
- I better understand the technology in my everyday life and can identify the programmatic concepts present in these technologies.

# Learning goals

- I am excited to use programming to solve real-world problems I encounter outside class, including those related to my major/career.
- I can break down complex problems into smaller subproblems by applying the logical reasoning skills I have gained from programming.
- I better understand the technology in my everyday life and can identify the programmatic concepts present in these technologies.
- I understand how computers process and organize information.

# Learning goals

- I am excited to use programming to solve real-world problems I encounter outside class, including those related to my major/career.

- I can break down complex problems into smaller subproblems by applying the logical reasoning skills I have gained from programming.

- I better understand the technology in my everyday life and can identify the programmatic concepts present in these technologies.

- I understand how computers process and organize information.

# Overarching questions

# Overarching questions

1. What is possible with technology and code?  What isn’t possible?

# Overarching questions

1. What is possible with technology and code?  What isn’t possible?

2. How can I use programming to solve problems that I otherwise would not be able to?

# Overarching questions

1. What is possible with technology and code?  What isn’t possible?

2. How can I use programming to solve problems that I otherwise would not be able to?

3. What makes “good” code and what are the best practices for writing code?

# Overarching questions

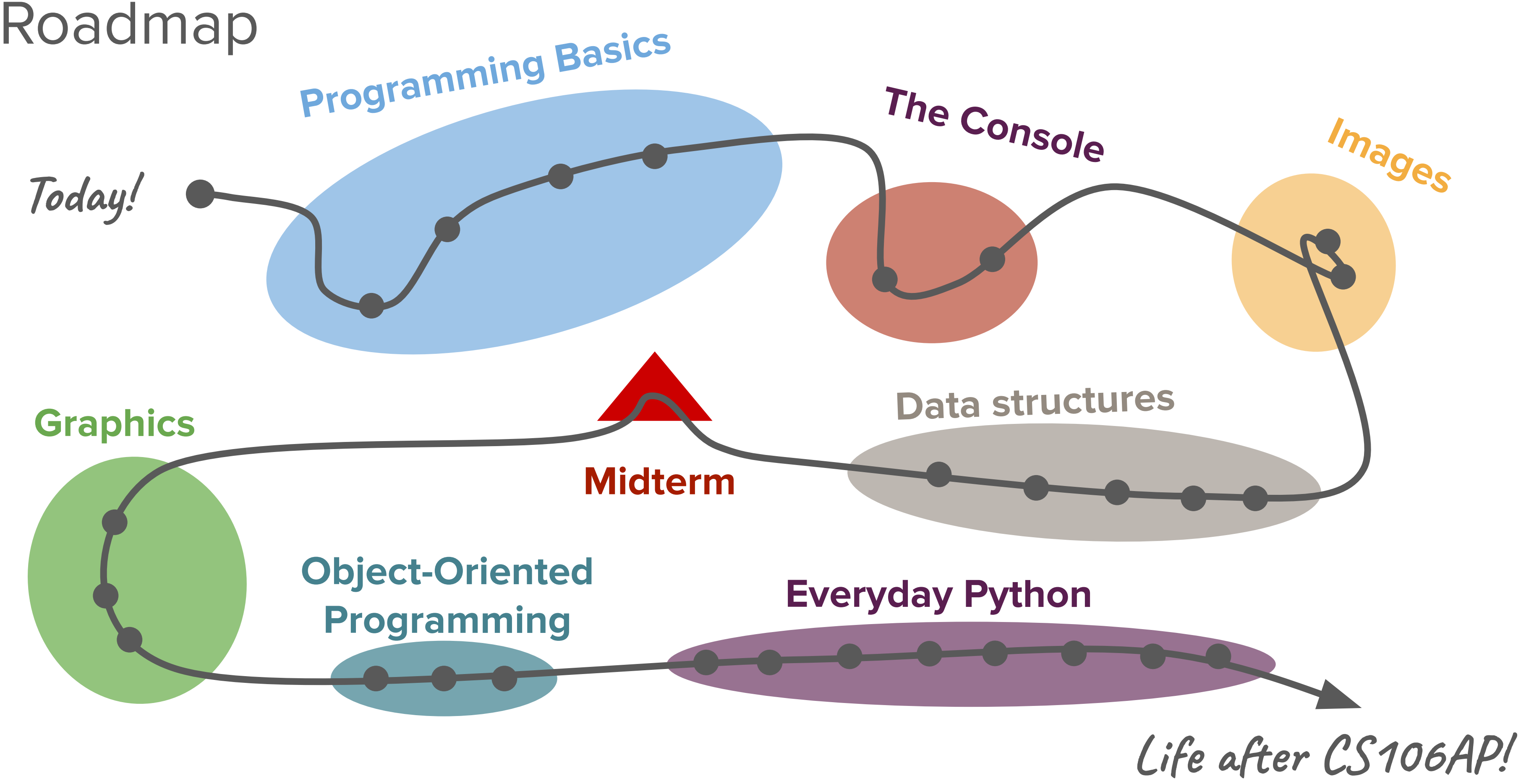
1. What is possible with technology and code?  What isn’t possible?

2. How can I use programming to solve problems that I otherwise would not be able to?

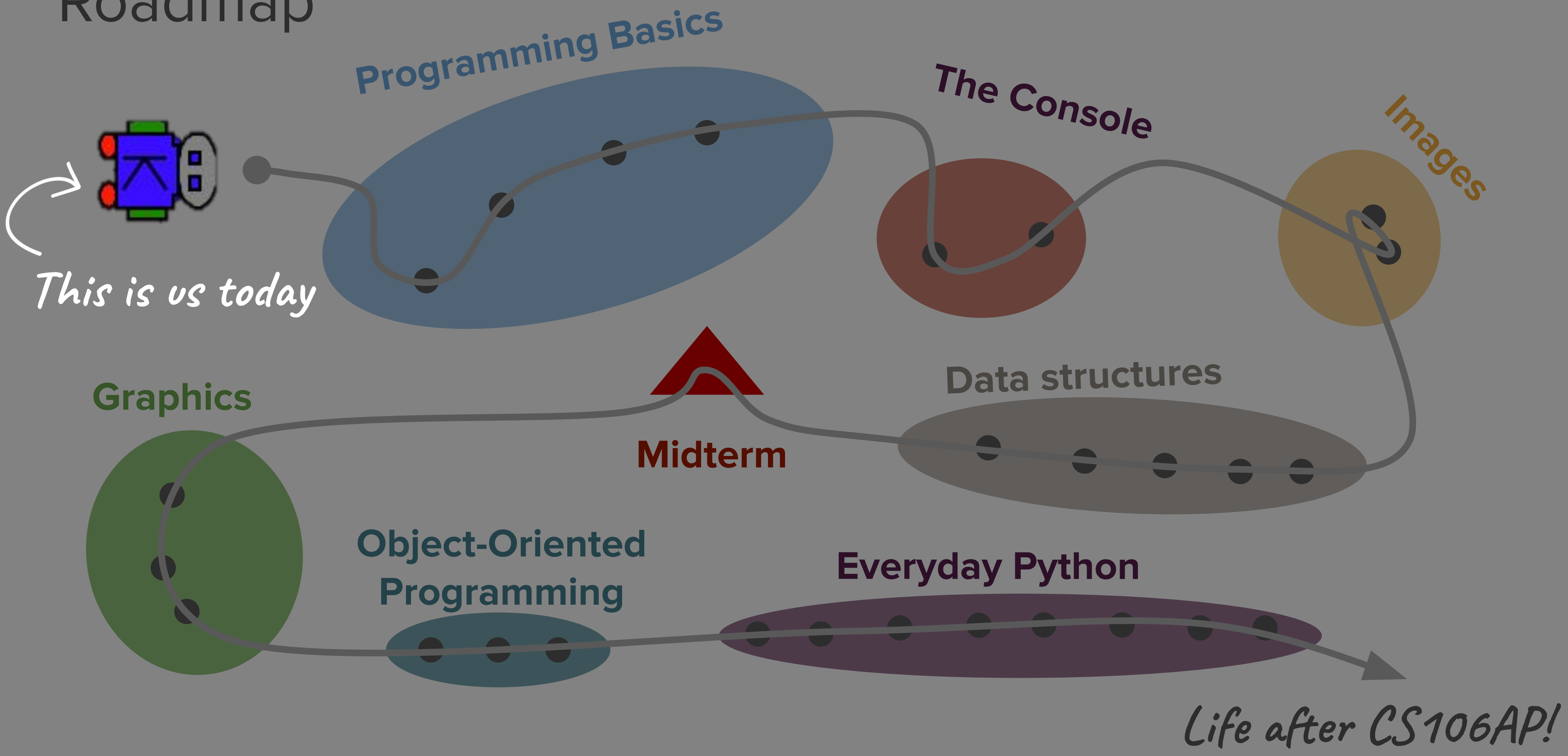
3. What makes “good” code and what are the best practices for writing code?

# Topic outline

# Roadmap


Programming Basics
Today!
The Console
Images
Graphics
Midterm
Data structures
Object-Oriented Programming
Everyday Python
Life after CS106AP!

# Roadmap



Programming Basics

The Console

Images

This is us today

Graphics

Midterm

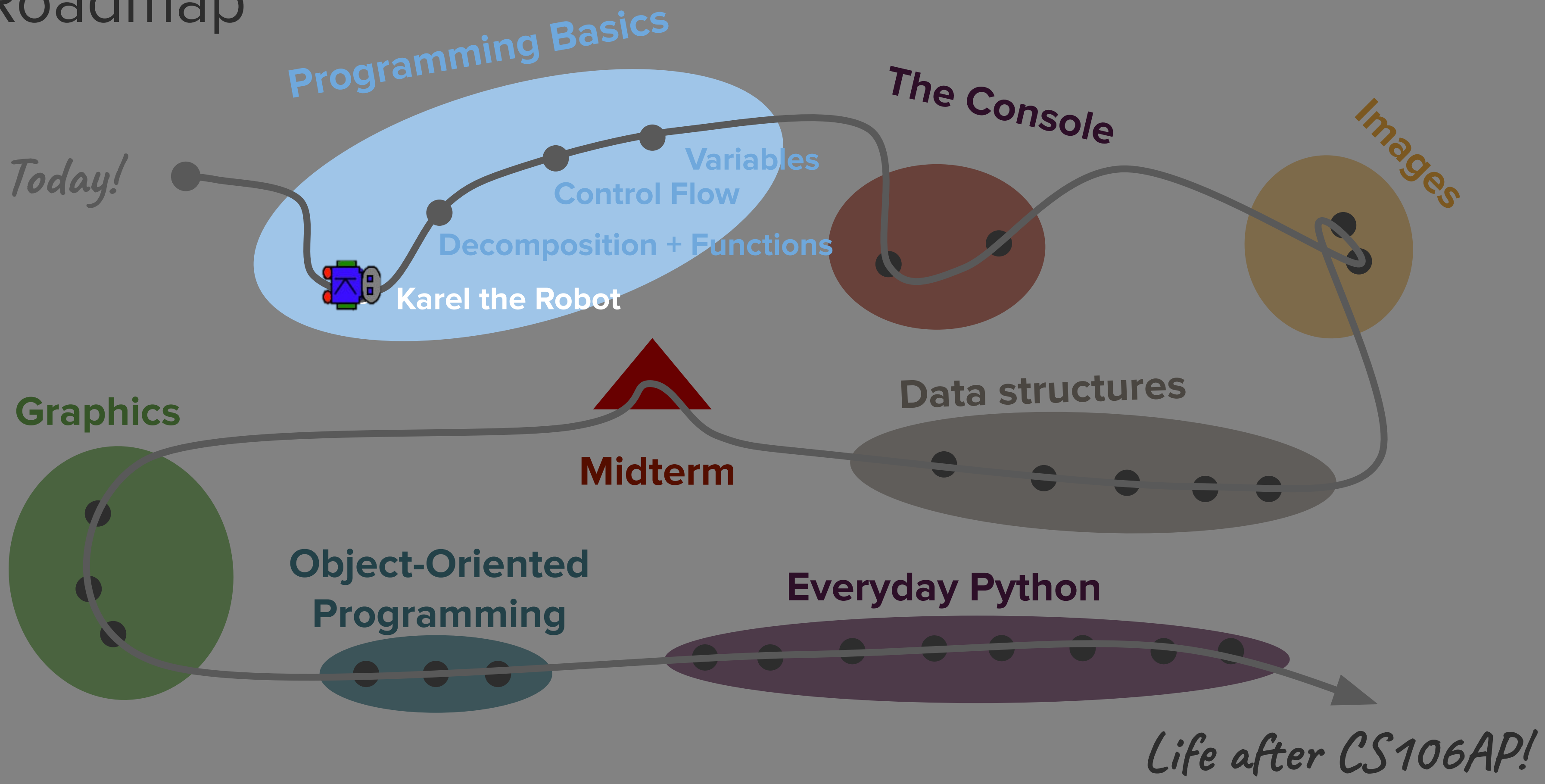
Data structures

Object-Oriented Programming

Everyday Python

Life after CS106AP!

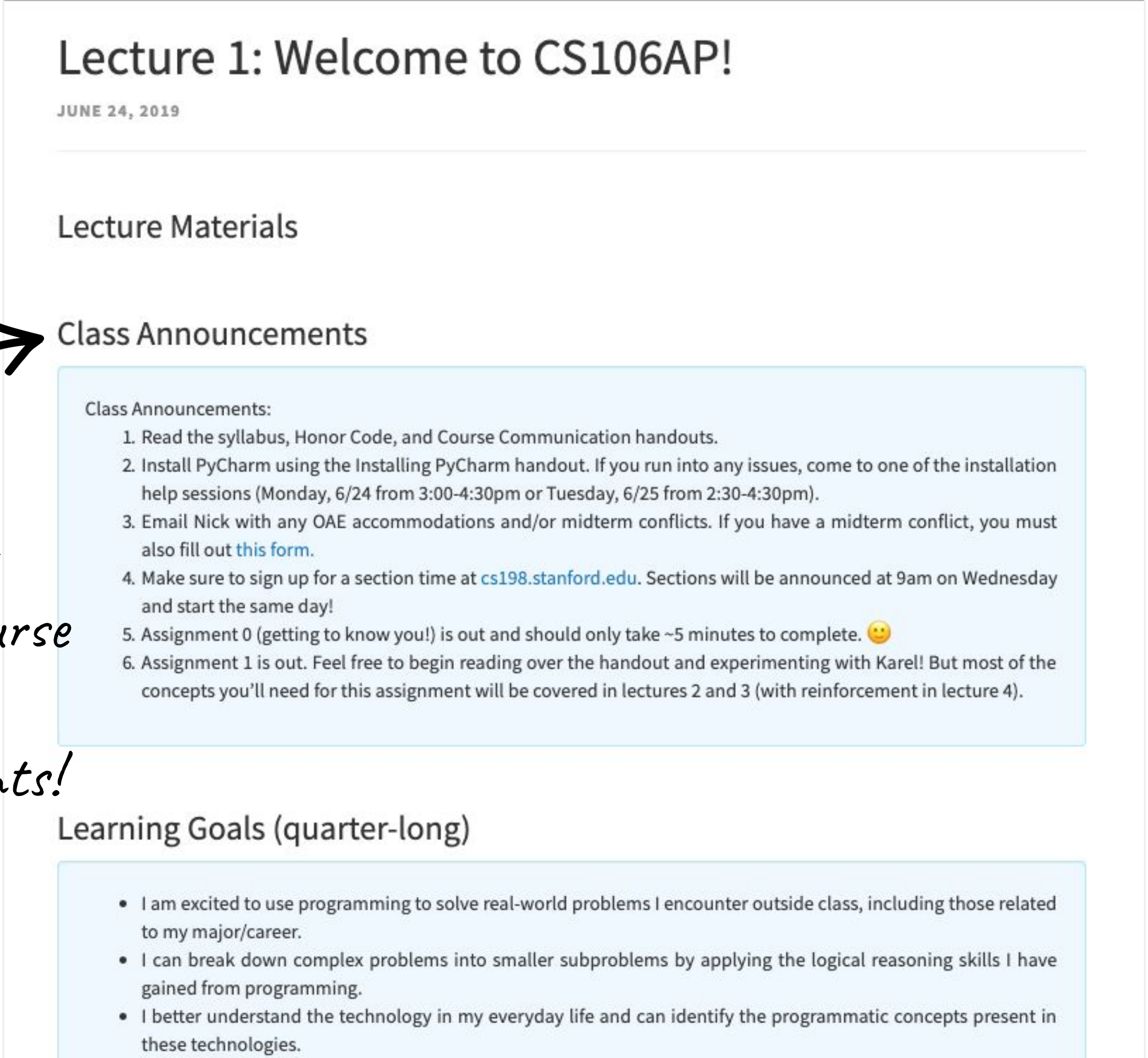
# Roadmap


Programming Basics
Today!
Variables
Control Flow
Decomposition + Functions
Karel the Robot
The Console
Images
Midterm
Data structures
Graphics
Object-Oriented Programming
Everyday Python
Life after CS106AP!

# What is CS106AP?

(Course logistics)

cs106ap.stanford.edu



# Lecture 1: Welcome to CS106AP!

JUNE 24, 2019

## Lecture Materials

## Class Announcements

Class Announcements:

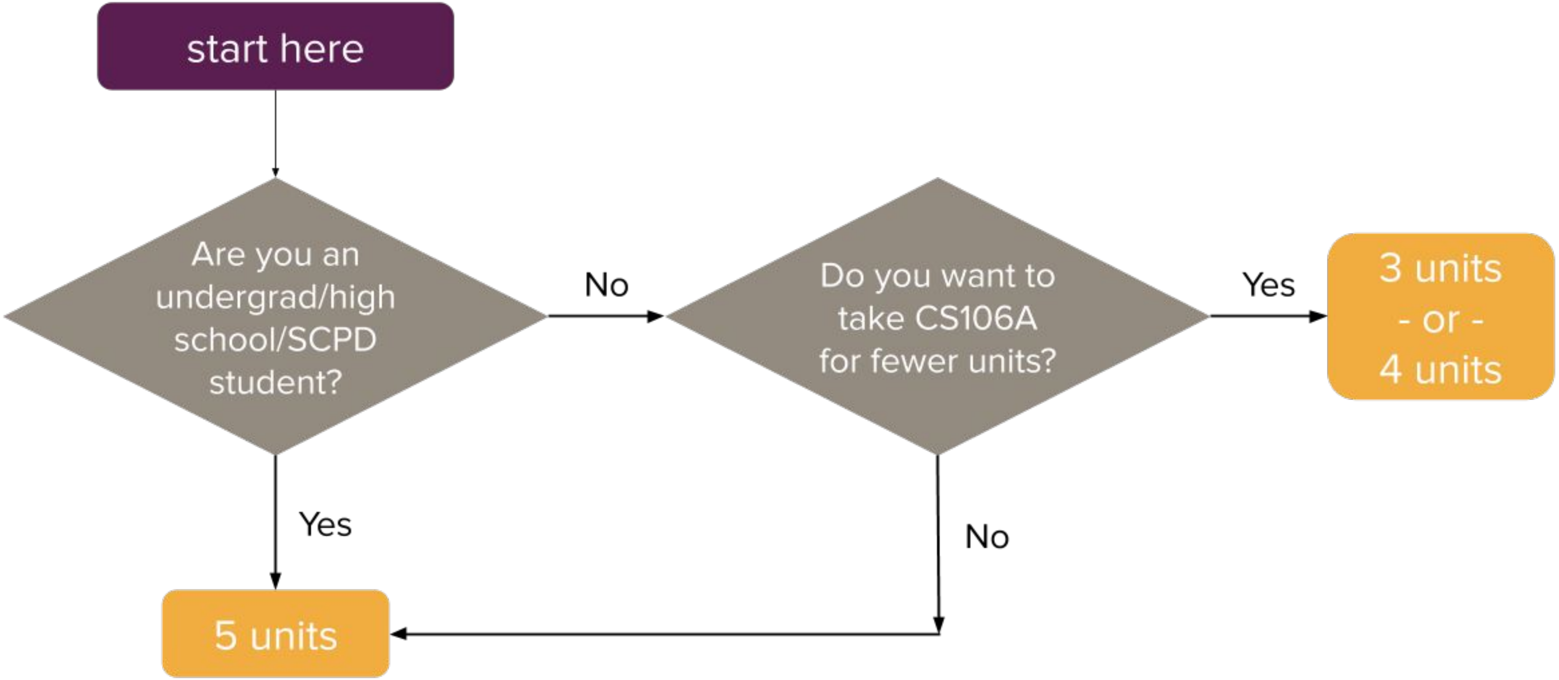
1. Read the syllabus, Honor Code, and Course Communication handouts.
2. Install PyCharm using the Installing PyCharm handout. If you run into any issues, come to one of the installation help sessions (Monday, 6/24 from 3:00-4:30pm or Tuesday, 6/25 from 2:30-4:30pm).
3. Email Nick with any OAE accommodations and/or midterm conflicts. If you have a midterm conflict, you must also fill out this form.
4. Make sure to sign up for a section time at cs198.stanford.edu. Sections will be announced at 9am on Wednesday and start the same day!
5. Assignment 0 (getting to know you!) is out and should only take ~5 minutes to complete. 🙂
6. Assignment 1 is out. Feel free to begin reading over the handout and experimenting with Karel! But most of the concepts you'll need for this assignment will be covered in lectures 2 and 3 (with reinforcement in lecture 4).

## Learning Goals (quarter-long)

- I am excited to use programming to solve real-world problems I encounter outside class, including those related to my major/career.
- I can break down complex problems into smaller subproblems by applying the logical reasoning skills I have gained from programming.
- I better understand the technology in my everyday life and can identify the programmatic concepts present in these technologies.

Make sure to check the course website for announcements!

# How many units?



start here
Are you an undergrad/high school/SCPD student?
Yes
5 units
No
Do you want to take CS106A for fewer units?
Yes
3 units - or - 4 units
No

*Diagram courtesy of Chris Piech*

# Why should I come to lecture?

# Lecture pedagogy

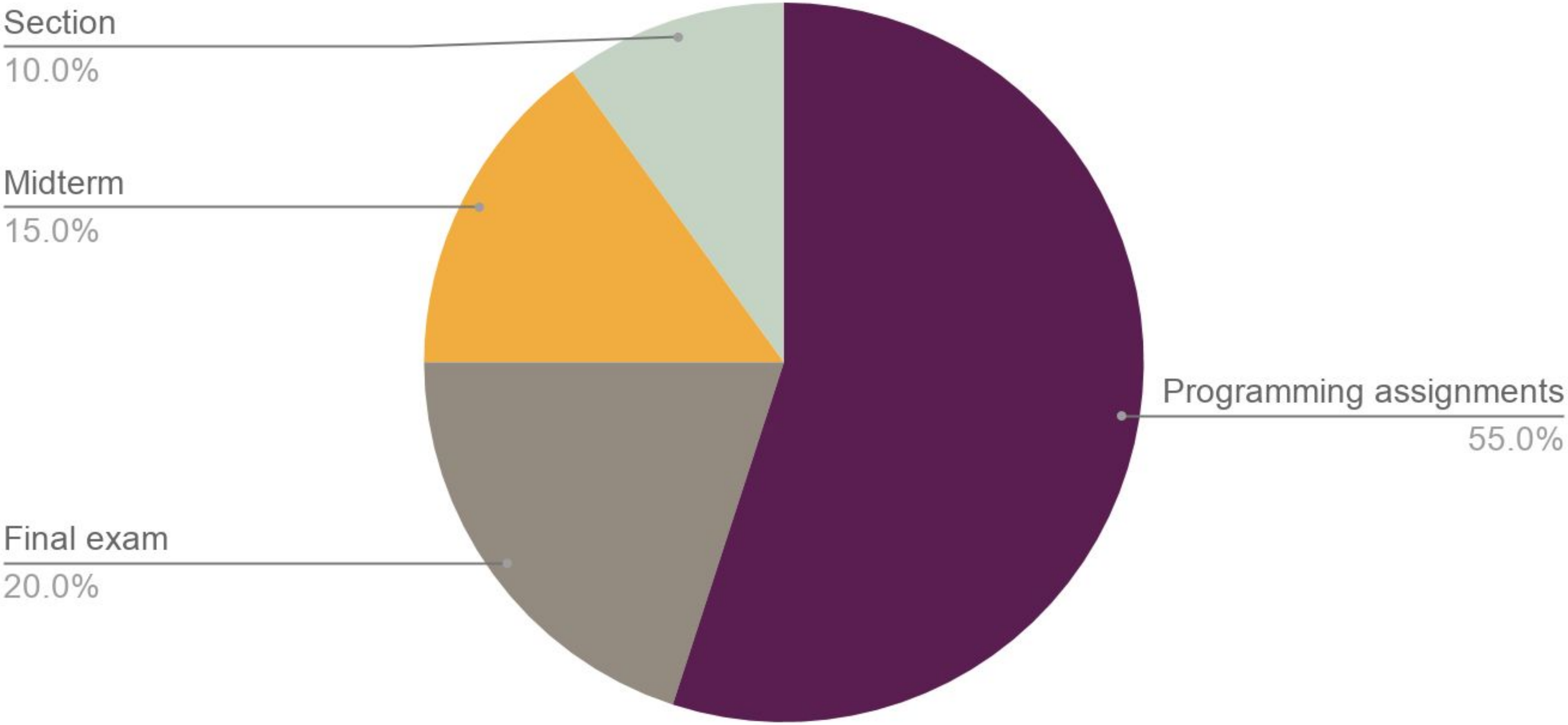
- Not just us talking at you: active learning exercises
  - Bring your laptop!

- Quick lecture-to-usage turnaround for concepts covered in class

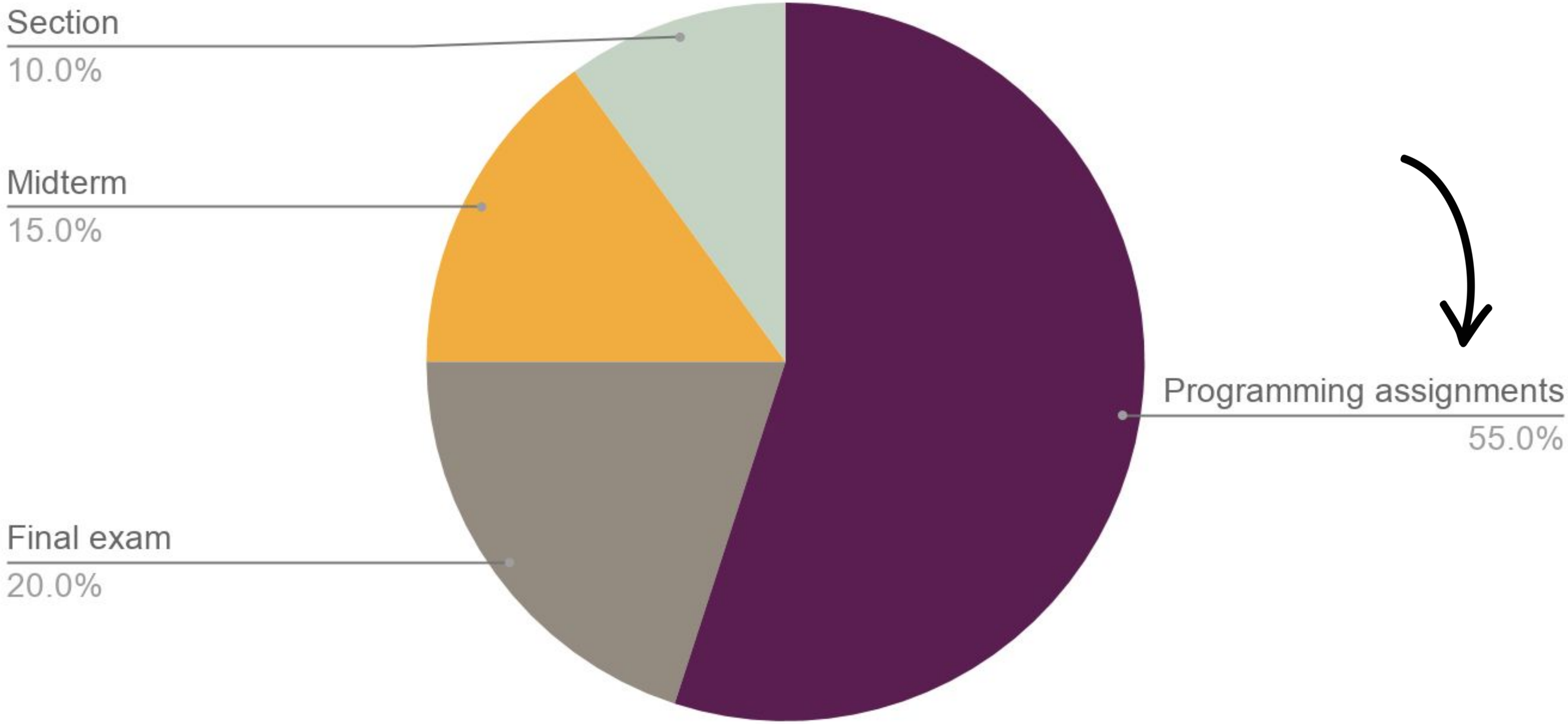
- We’ll stick around to answer questions afterward!

# How will I be assessed?

# What we will ask you to do



# What we will ask you to do



# Programming assignments

- There will be 6 total
    - First three focus on practice with CS fundamentals
    - Last three focus on CS applications

# Programming assignments

- There will be 6 total
- Graded on **functionality** and **style** using buckets

| | |
|---|---|
| ✓ | Meets requirements, possibly with a few small problems |

# Programming assignments

- There will be 6 total
- Graded on **functionality** and **style** using buckets

| | |
|---|---|
| ✓+ | Satisfies all requirements for the assignment |
| ✓ | Meets requirements, possibly with a few small problems |
| ✓- | Has problems serious enough to fall short of requirements |

# Programming assignments

- There will be 6 total
- Graded on **functionality** and **style** using buckets

| | |
|---|---|
| ++ | Absolutely fantastic submission (extremely rare) |
| + | "Perfect" or exceeds our standard expectations |
| ✓+ | Satisfies all requirements for the assignment |
| ✓ | Meets requirements, possibly with a few small problems |
| ✓- | Has problems serious enough to fall short of requirements |
| - | Extremely serious problems, but shows some effort |
| -- | Shows little effort and does not represent passing work |

# Programming assignments

- There will be 6 total
- Graded on **functionality** and **style** using **buckets**

Why?

| | |
|---|---|
| ++ | Absolutely fantastic submission (extremely rare) |
| + | "Perfect" or exceeds our standard expectations |
| ✓+ | Satisfies all requirements for the assignment |
| ✓ | Meets requirements, possibly with a few small problems |
| ✓- | Has problems serious enough to fall short of requirements |
| - | Extremely serious problems, but shows some effort |
| -- | Shows little effort and does not represent passing work |

# Programming assignments

- There will be 6 total
- Graded on functionality and style using buckets
- You can submit revisions if you receive below a check
  - Must be turned in up to three days after the next assignment is due.
  - We want to give you opportunities to demonstrate learning!
  - The revisions must include the updated code, tests to catch previous errors, and must not introduce new errors.
  - Grade capped at a check.

# Programming assignments

- There will be 6 total
- Graded on functionality and style using buckets
- You can submit revisions if you receive below a check
- Two deadlines for each assignment
  - Aim for the first, “on-time” deadline! (5% bonus)
  - The second, “extended” deadline is like a free 24-hour extension – only take it if you have a particularly difficult week.
  - **Late policy**: Anything after the second deadline will drop a bucket for every 24-hour window, and anything later than 48 hours after the “extended” deadline will get a 0.

# Programming assignments

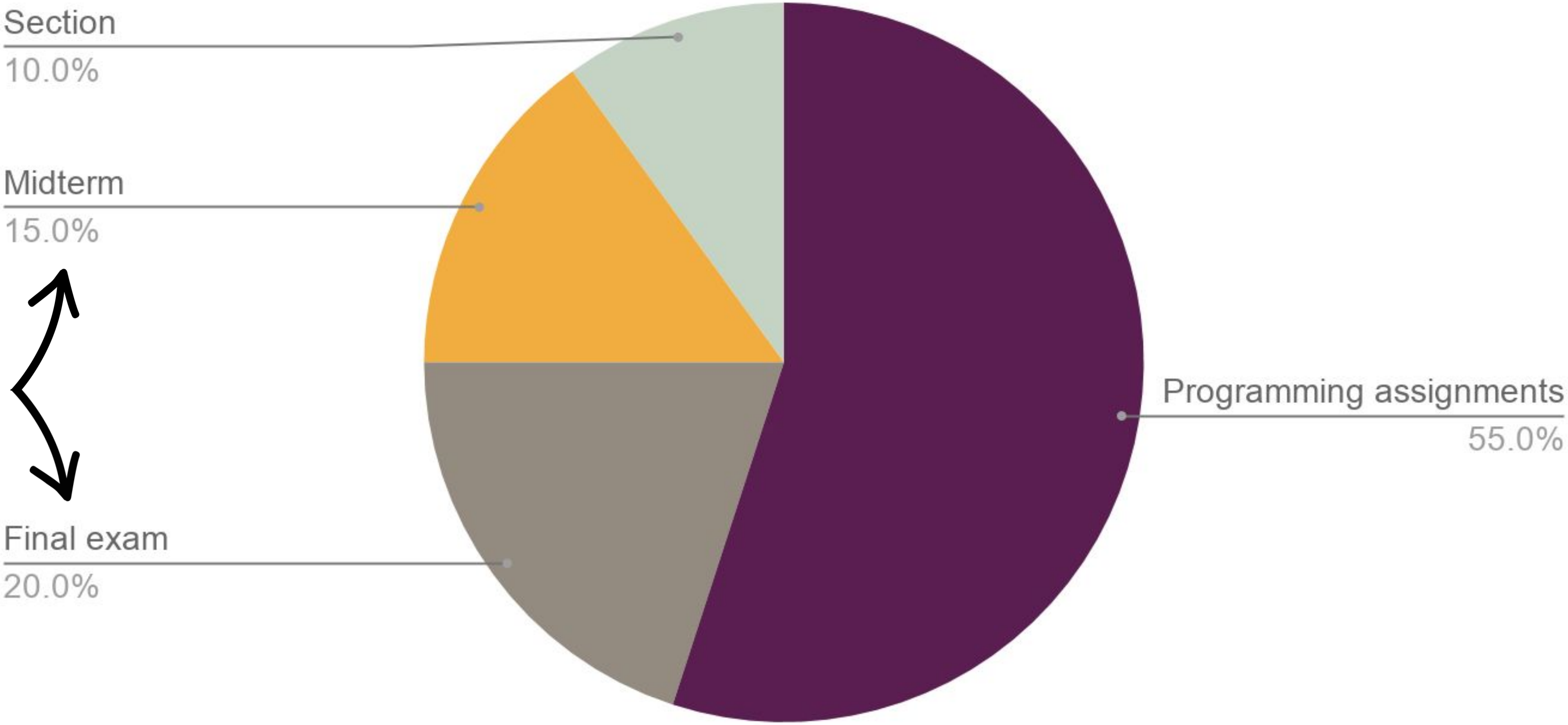
- There will be 6 total
- Graded on functionality and style using buckets
- You can submit revisions if you receive below a check
- Two deadlines for each assignment

# Programming assignments

- There will be 6 total
- Graded on functionality and style using buckets
- You can submit revisions if you receive below a check
- Two deadlines for each assignment

All deadlines are at **11:59pm** (including for revisions).

# What we will ask you to do



# Exams

- Midterm: Monday, July 22 from 7:00pm-9:00pm
    - If you have an academic/immovable conflict, please email Nick + fill out the conflict form by July 15.

- Final: Friday, August 16 from 3:30pm-6:30pm
    - **No alternates** (except OAE accommodations)

- If you have OAE accommodations, please email Nick.
- Locations TBD
- 10 pages of notes

# Exams

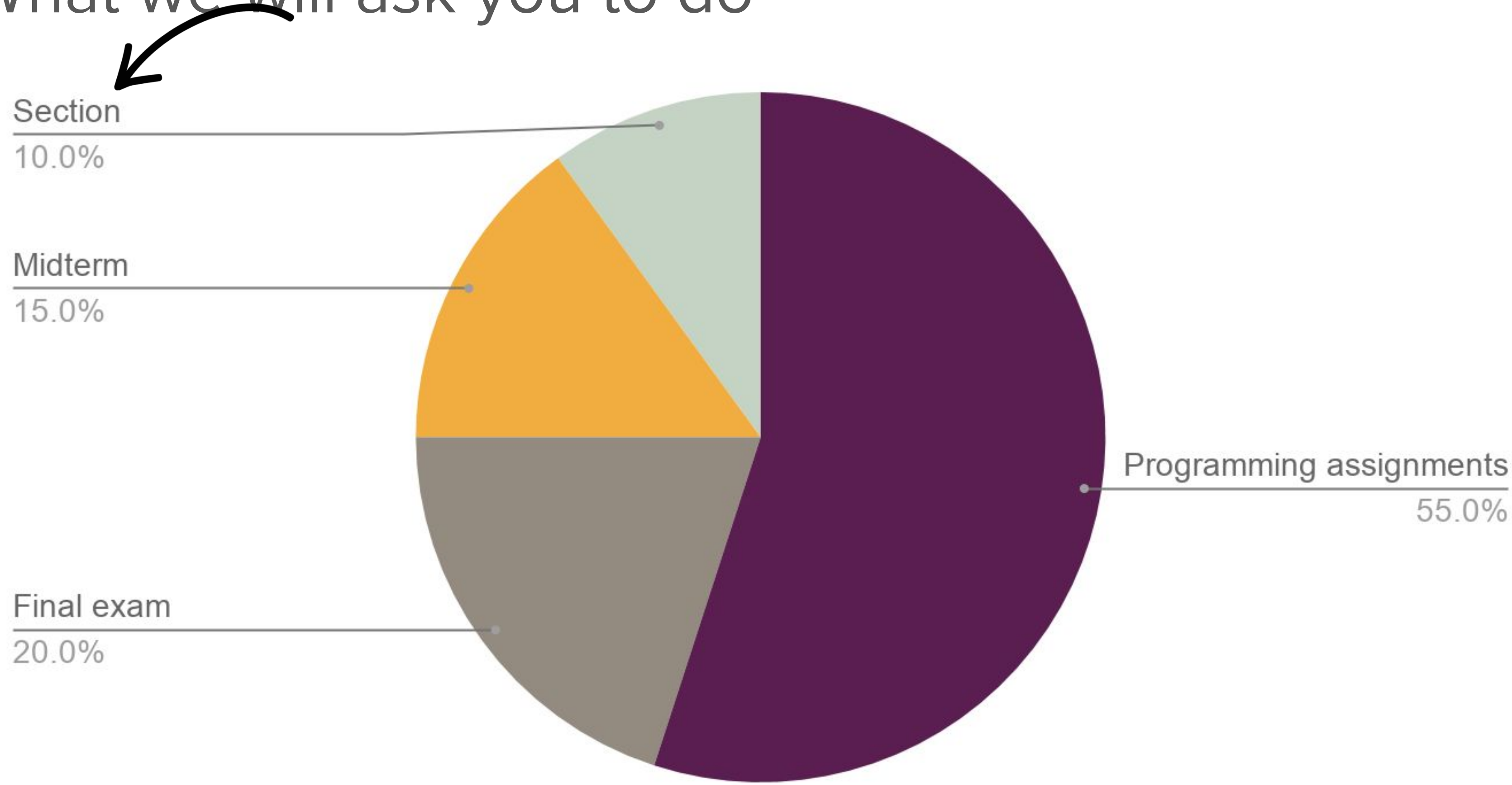
- Midterm: Monday, July 22 from 7:00pm-9:00pm
  - If you have an academic/immovable conflict, please email Nick + fill out the conflict form by July 15.

- Final: Friday, August 16 from 3:30pm-6:30pm
  - **No alternates** (except OAE accommodations)

- If you have OAE accommodations, please email Nick.
- Locations TBD
- 10 pages of notes

Why?

# What we will ask you to do



# Section

- Sign up by **Tuesday at 5pm** at cs198.stanford.edu
    - Sections with remaining spots will open for signups after Wednesday at 9am

# Section

- Sign up by Tuesday at 5pm at cs198.stanford.edu

- Partners must be in the same section
  - You must submit the exact same section preferences to guarantee staying together
  - Partners will only be allowed on Assignment 4 and onwards
  - Both partners get the same grade on an assignment

# Section

- Sign up by Tuesday at 5pm at cs198.stanford.edu
- Partners must be in the same section
- Sections start this Wednesday!

# Section

- Sign up by Tuesday at 5pm at [cs198.stanford.edu]
- Partners must be in the same section
- Sections start this Wednesday!

# How do I get help?

# What the course staff do

- Clarify conceptual material
- Help you develop good debugging practices
- Answer any administrative questions
- Chat about CS and life in general!

# What the course staff do

- Clarify conceptual material
- Help you develop good debugging practices
- Answer any administrative questions
- Chat about CS and life in general!

We're always happy to help you apply CS and the concepts you've learned in class to real-world applications/areas you're interested in.

# What the course staff **don’t** do

- Write your code for you

- Solve your bugs on assignments

# What the course staff **don’t** do

- Write your code for you

- Solve your bugs on assignments

*This is how you learn as a student!*

# Resources for getting help

- LaIR (general office hours)
    - Open Sunday through Wednesday, 7pm-11pm at Tresidder 1st floor
    - Starts this Wednesday
- Your section leader
- Nick’s office hours
- Kylie’s + Sonja’s office hours
- Piazza

# Resources for getting help

- LaIR
- Your section leader
- Nick’s office hours
- Kylie’s + Sonja’s office hours
- Piazza

# Resources for getting help

- **LaIR**
- **Your section leader**
- **Nick's office hours**
- **Kylie's + Sonja's office hours**
- **Piazza**

Conceptual question?

# Resources for getting help

- **(C)LaIR**
- **Your section leader**
- **Nick's office hours**
- **Kylie's + Sonja's office hours**
- **Piazza**

Conceptual question?

# Resources for getting help

- **LaIR**
- **Your section leader**
- **Nick’s office hours**
- Kylie’s + Sonja’s office hours
- Piazza

Debugging help + code questions?

# Resources for getting help

- LaIR
- Your section leader
- **Nick's ~~office hours~~ email**
- Kylie's + Sonja's office hours
- **Piazza**

Administrative questions?

# Resources for getting help

- LaIR
- **Your section leader**
- **Nick's office hours**
- **Kylie's + Sonja's office hours**
- Piazza

General CS + life questions?

# Resources for getting help

- LaIR
- Your section leader
- Nick’s office hours
- Kylie’s + Sonja’s office hours
- Piazza

When in doubt, check the Course Communication guidelines!

# Resources for getting help

- LaIR
- Your section leader
- Nick’s office hours
- Kylie’s + Sonja’s office hours
- Piazza

When in doubt, check the Course Communication guidelines!

---

The Summer Academic Resource Center (SARC) also offers non-CS tutoring and academic workshops separate from our course.

# Honor Code

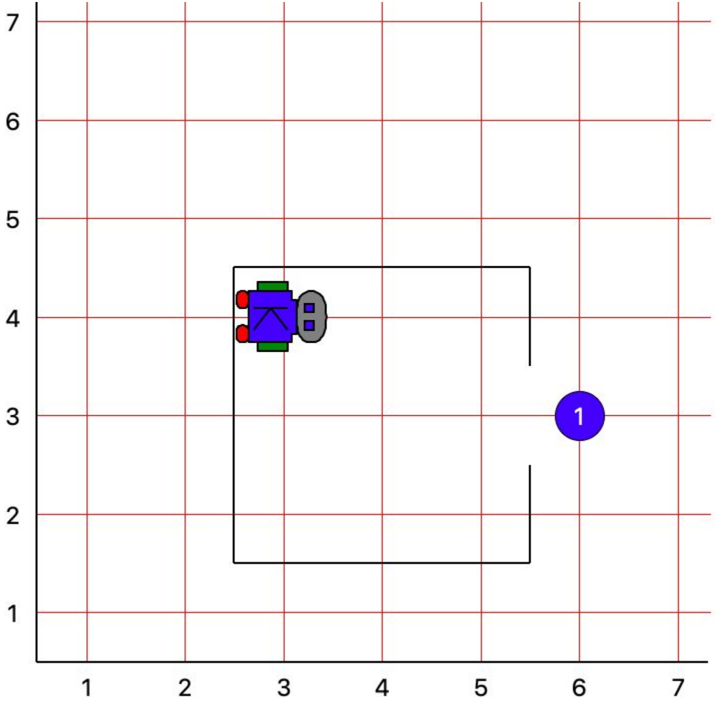
# Stanford's Honor Code

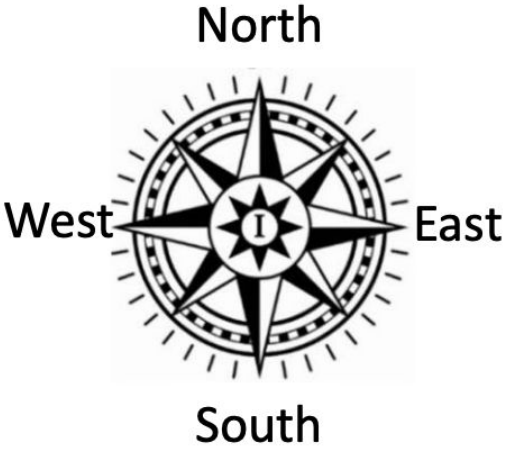
- All students in the course must abide by the **Stanford Honor Code**.
- Make sure to read over the **Honor Code handout** on the CS106AP website for CS-specific expectations.
- Acknowledge any help you get outside course staff directly in your submissions.
- We run code similarity software on all of your programs.
- Anyone caught violating the Honor Code will automatically fail the course.

# Who is Karel?

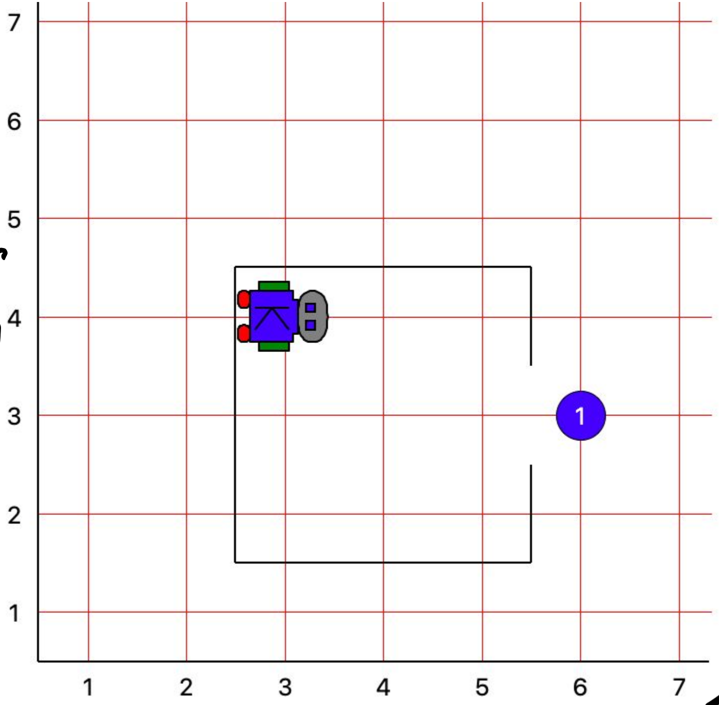
# Meet Karel!

# This is Karel's world



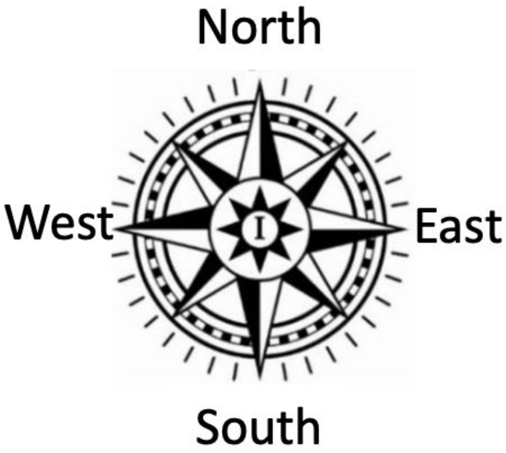


# This is Karel's world

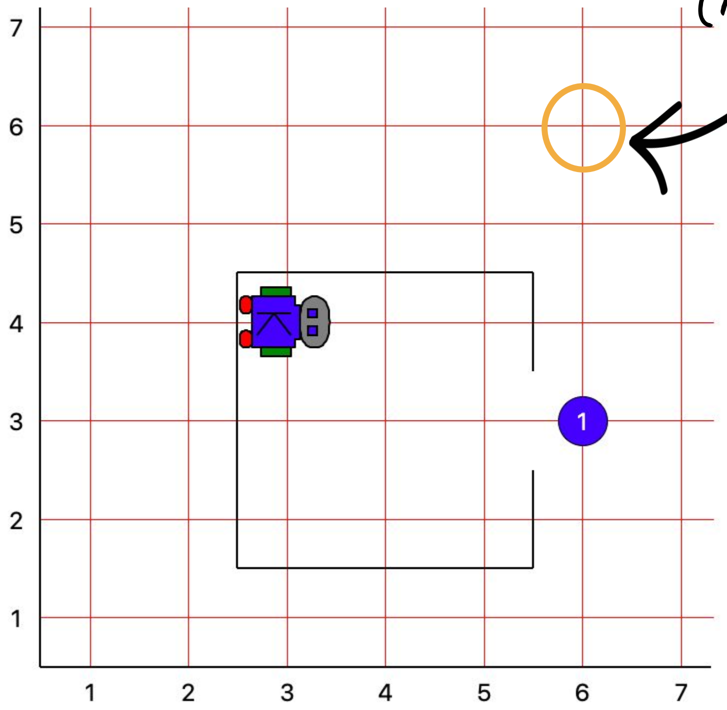


streets
(rows)

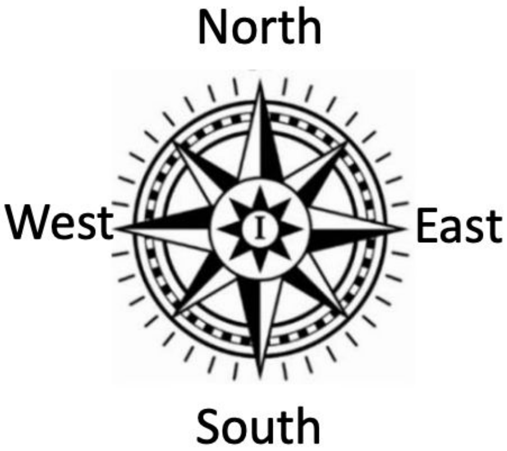
avenues
(columns)



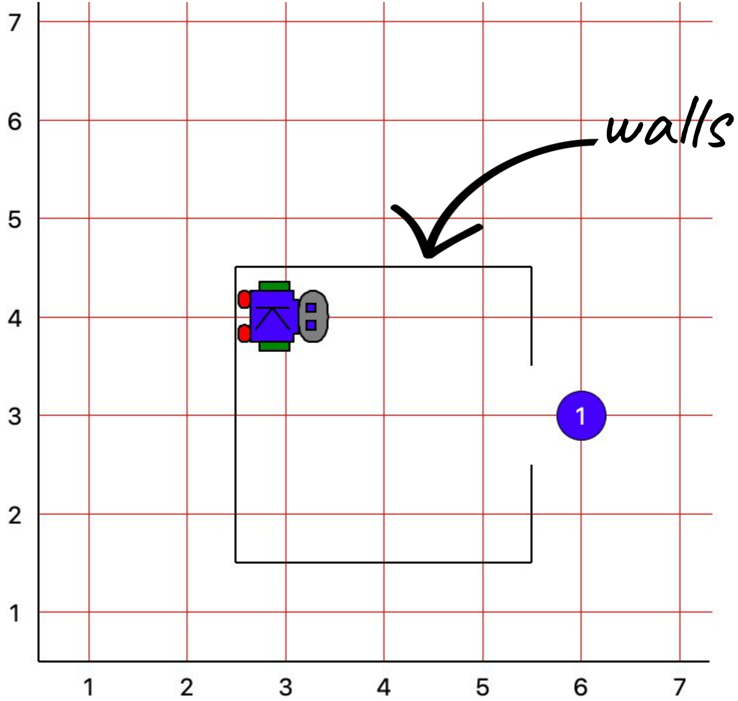
# This is Karel's world

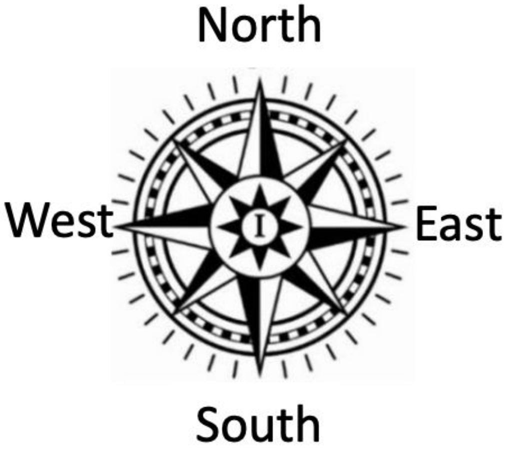


corners
(intersections)

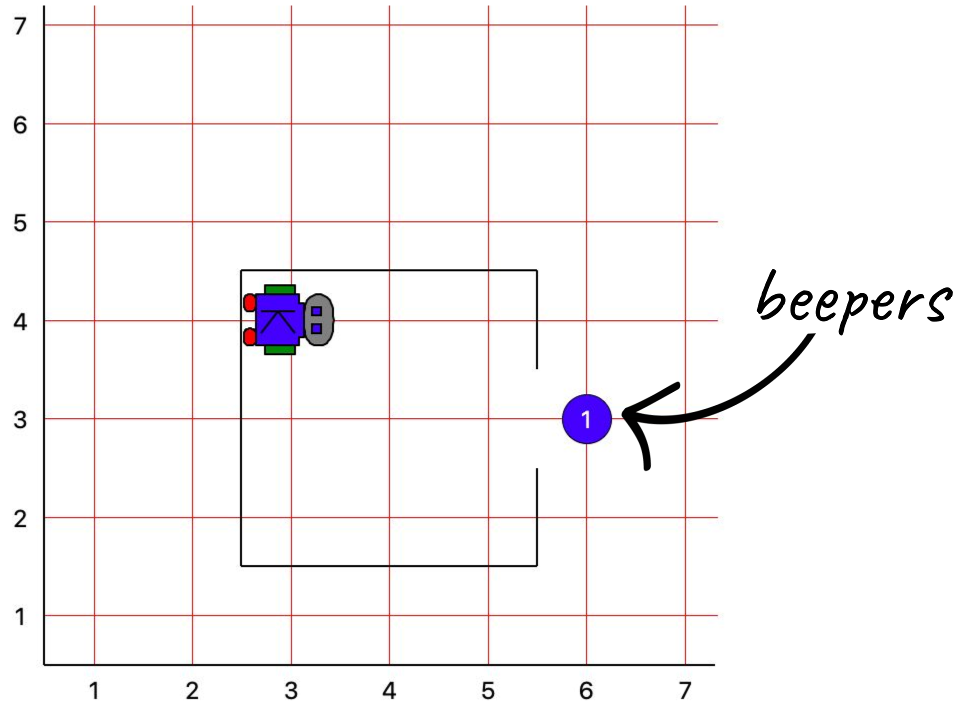


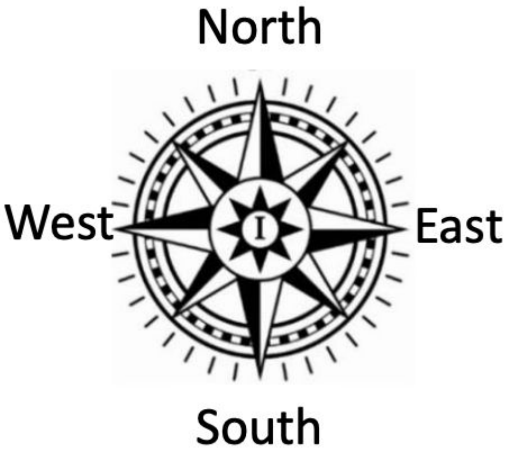
# This is Karel's world



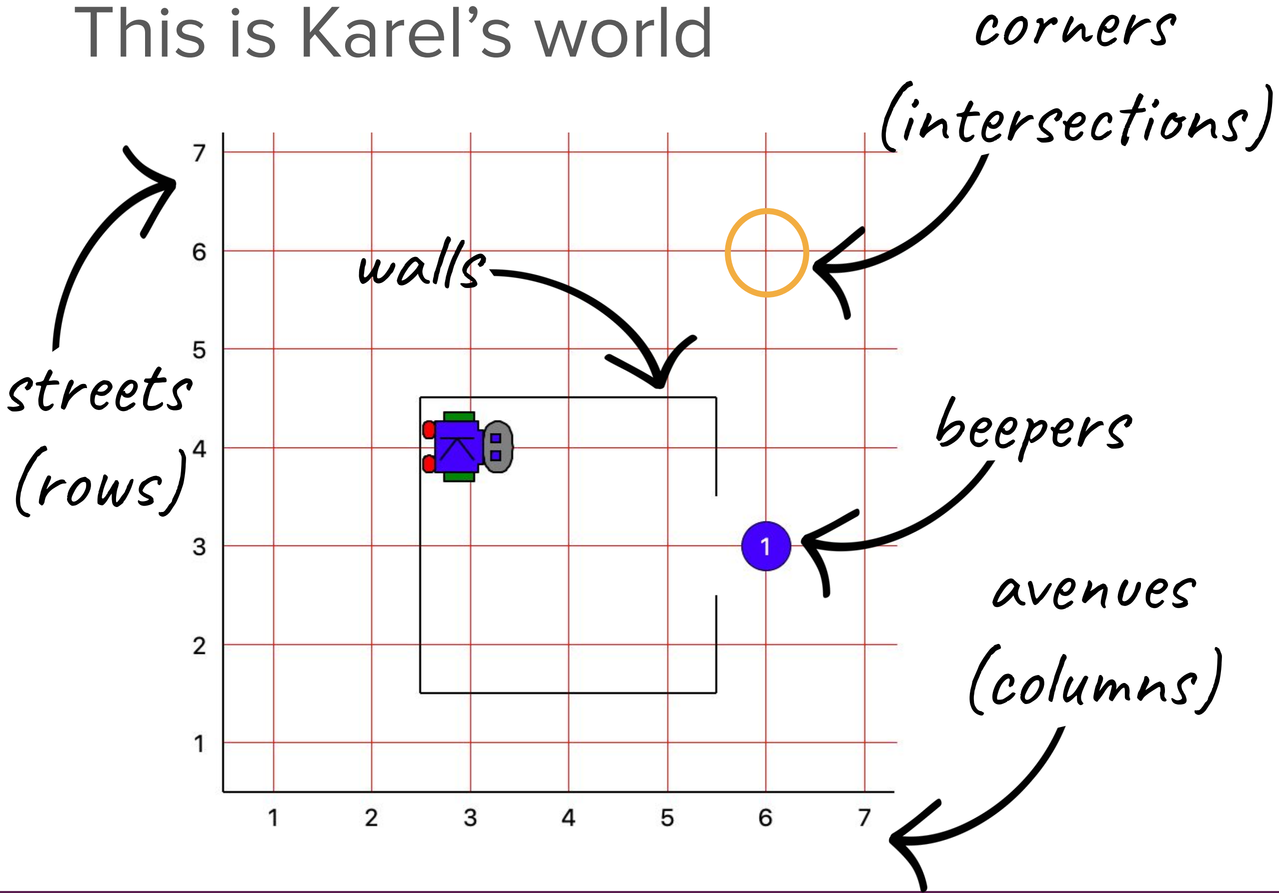


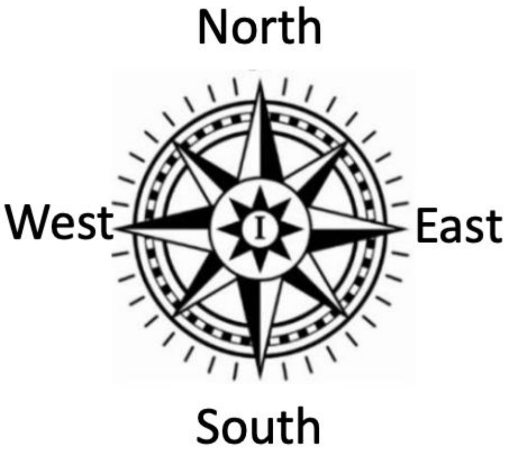
# This is Karel’s world



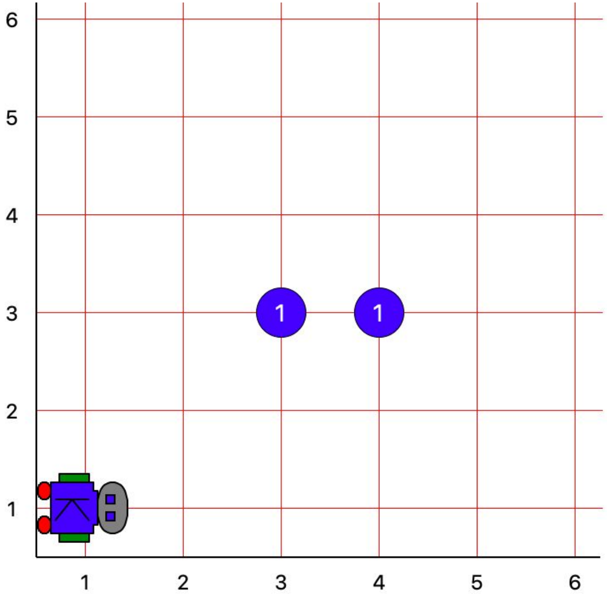


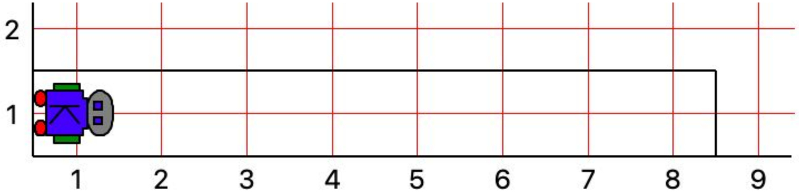
# This is Karel's world





# Karel's world can look different

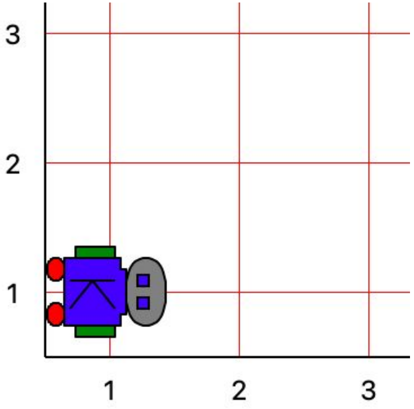




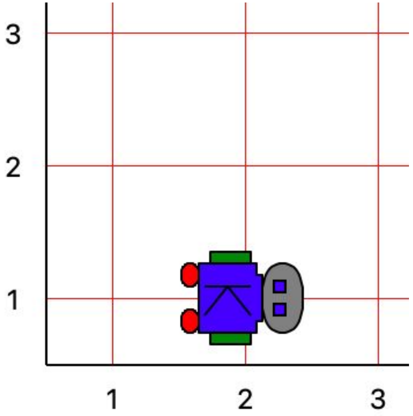
# What Karel can do

```
move()

turn_left()

put_beeper()

pick_beeper()
```

# What Karel can do



begin



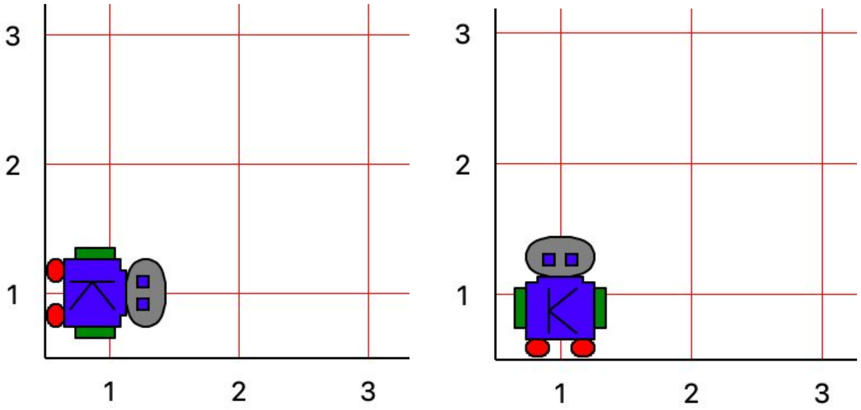
end

**`move()`**

Makes Karel move forward one square in the direction it is facing.

Errors if there is a wall in front of Karel.

# What Karel can do



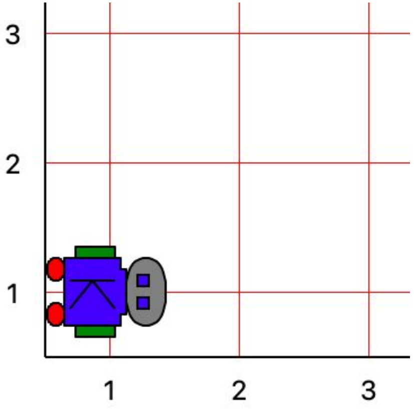
begin

end

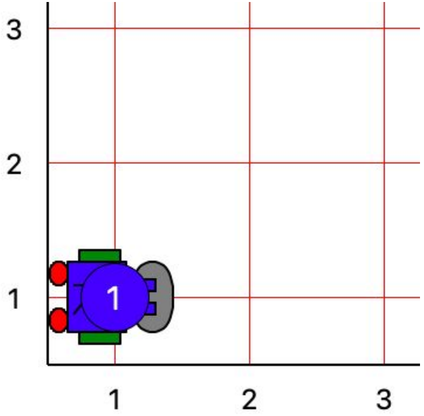
`turn_left()`

Makes Karel turn left.

# What Karel can do



begin



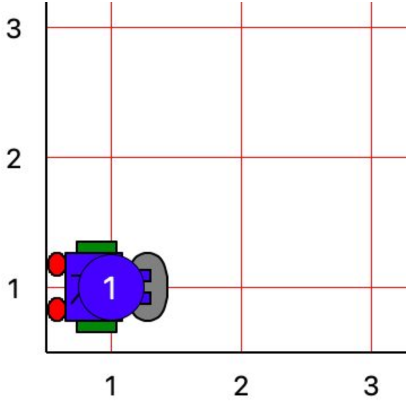
end

## `put_beeper()`

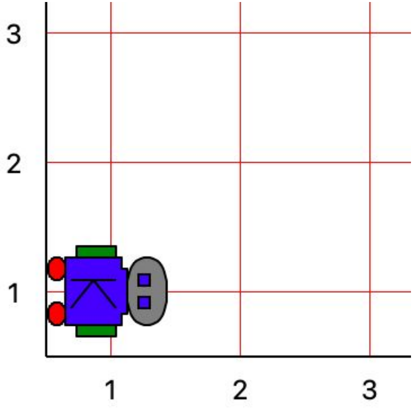
Makes Karel place a beeper on the corner where it is currently standing.

You can place multiple beepers on any given corner.

# What Karel can do



begin



end

## `pick_beeper()`

Makes Karel pick up a beeper from the corner where it is currently standing.

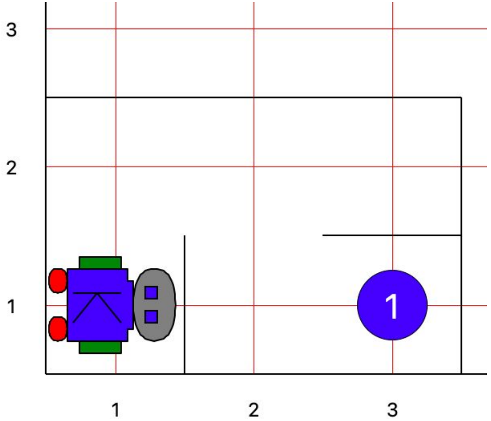
Errors if there are no beepers present on the corner.

# What Karel can do

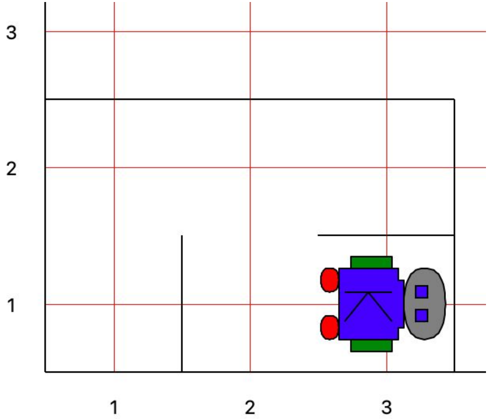
```
move()

turn_left()

put_beeper()

pick_beeper()
```

# Karel demo!

# Karel Goes Home



begin



end

# What **you** will do with Karel

- We write Karel code in a code editor called PyCharm.
  - Instructions for downloading and using PyCharm are on the course website
  - Please try doing this tonight!

# What **you** will do with Karel

- We write Karel code in a code editor called PyCharm.
  - Instructions for downloading and using PyCharm are on the course website
  - Please try doing this tonight!

**PyCharm installation help sessions:**

TODAY from 3-4:30pm in Gates 200

Tuesday, 6/25 from 2:30-4:30pm in Gates B02

# What **you** will do with Karel

- We write Karel code in a code editor called PyCharm.
    - Instructions for downloading and using PyCharm are on the course website
    - Please try doing this tonight!

- Assignment 1 will be released tomorrow
    - You'll learn more about programming with Karel later this week to complete the assignment!

# What’s next?

# Who are you?

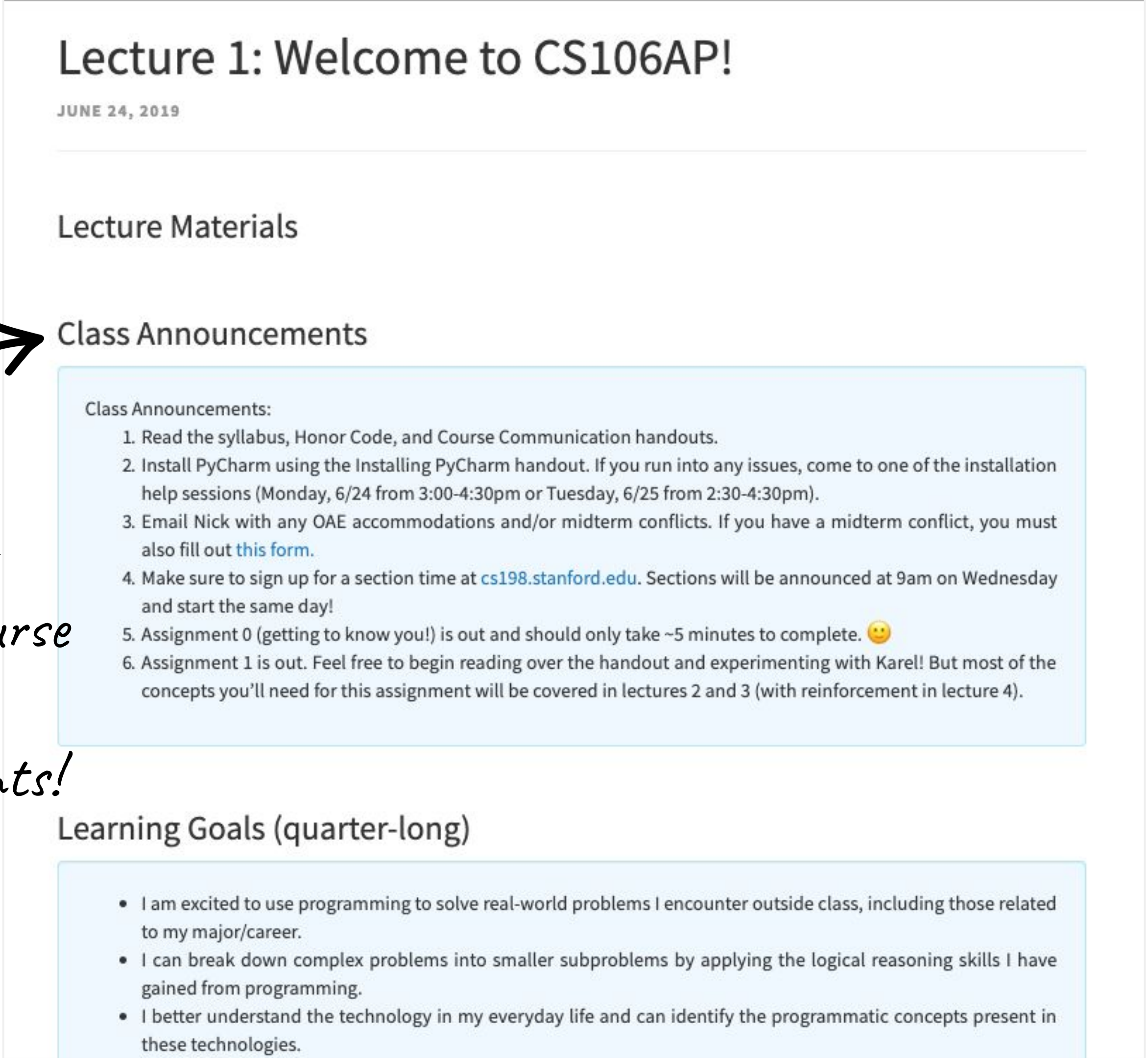
(Assignment 0)

due this Wednesday, 6/26 at 11:59pm!

# Programming with Karel

(Assignment 1)

due next Tuesday, 7/2 at 11:59pm!



# Lecture 1: Welcome to CS106AP!

JUNE 24, 2019

## Lecture Materials

## Class Announcements

Class Announcements:

1. Read the syllabus, Honor Code, and Course Communication handouts.
2. Install PyCharm using the Installing PyCharm handout. If you run into any issues, come to one of the installation help sessions (Monday, 6/24 from 3:00-4:30pm or Tuesday, 6/25 from 2:30-4:30pm).
3. Email Nick with any OAE accommodations and/or midterm conflicts. If you have a midterm conflict, you must also fill out this form.
4. Make sure to sign up for a section time at cs198.stanford.edu. Sections will be announced at 9am on Wednesday and start the same day!
5. Assignment 0 (getting to know you!) is out and should only take ~5 minutes to complete. 🙂
6. Assignment 1 is out. Feel free to begin reading over the handout and experimenting with Karel! But most of the concepts you'll need for this assignment will be covered in lectures 2 and 3 (with reinforcement in lecture 4).

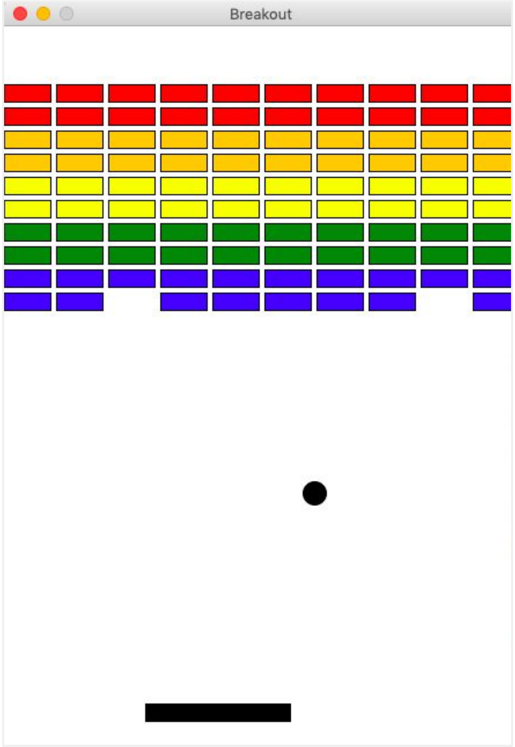
## Learning Goals (quarter-long)

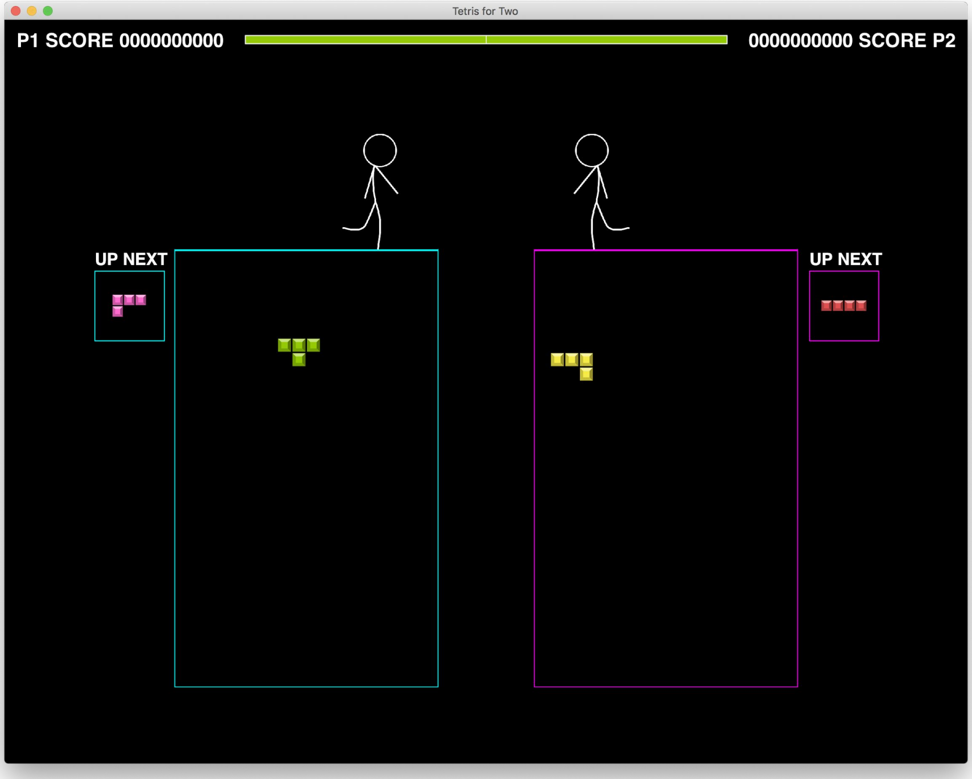
- I am excited to use programming to solve real-world problems I encounter outside class, including those related to my major/career.
- I can break down complex problems into smaller subproblems by applying the logical reasoning skills I have gained from programming.
- I better understand the technology in my everyday life and can identify the programmatic concepts present in these technologies.

Make sure to check the course website for announcements!

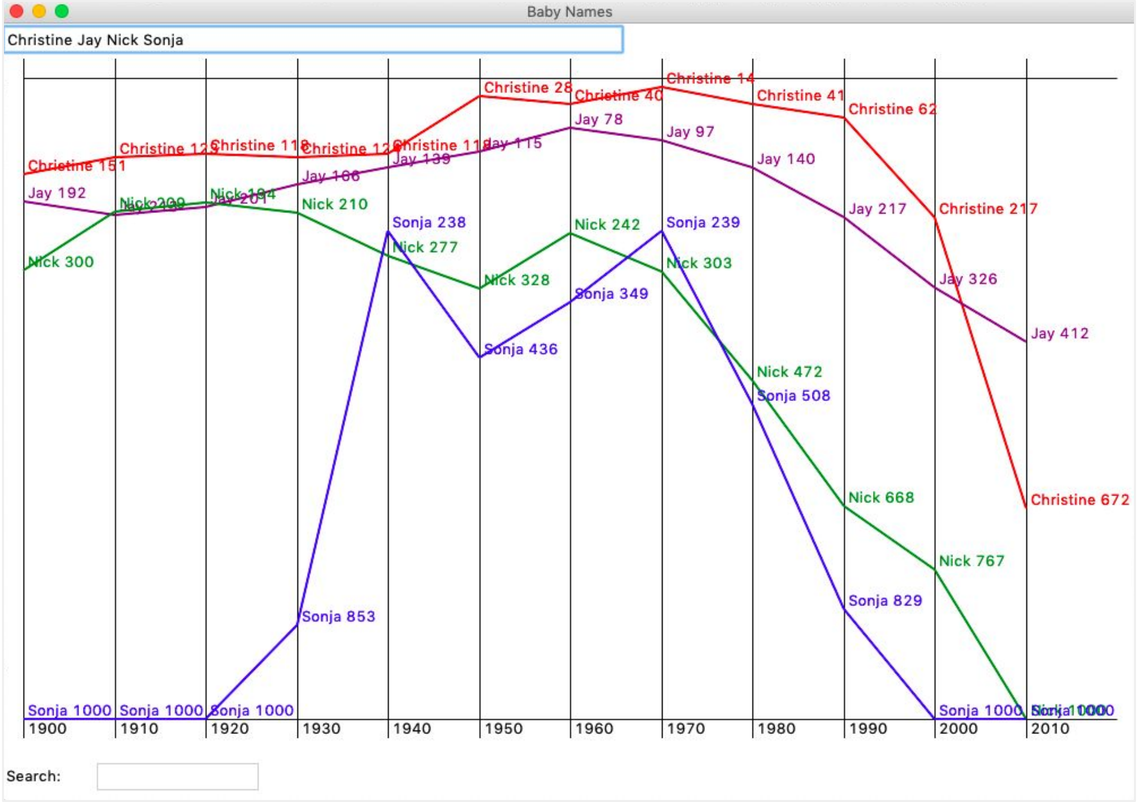
# Applications of CS

# Games


Breakout


Tetris for Two
P1 SCORE 0000000000
0000000000 SCORE P2
UP NEXT
UP NEXT

# Data visualization

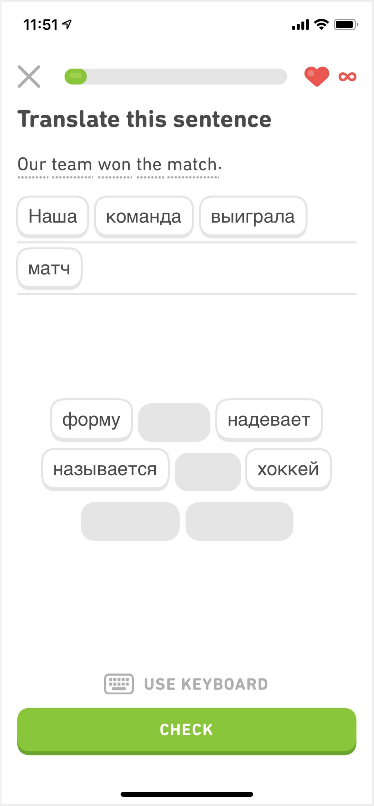


Baby Names

Christine Jay Nick Sonja

Search:

# Web and mobile apps

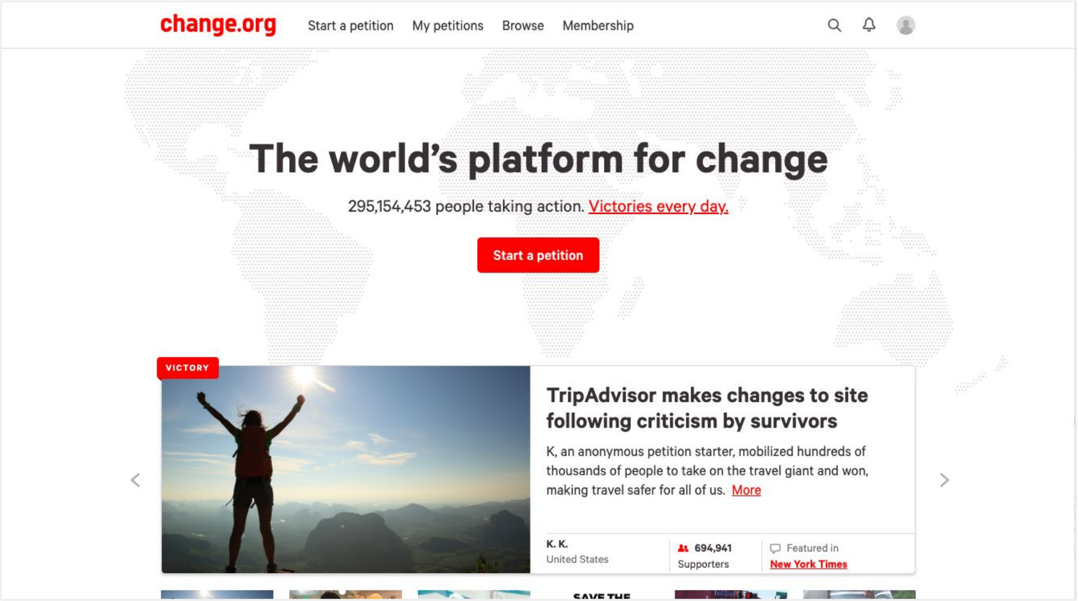


11:51

Translate this sentence

Our team won the match.

Наша команда выиграла матч



The world's platform for change

295,154,453 people taking action. Victories every day.

TripAdvisor makes changes to site following criticism by survivors

K, an anonymous petition starter, mobilized hundreds of thousands of people to take on the travel giant and won, making travel safer for all of us. More

K. K.
United States

694,941
Supporters

Featured in
New York Times

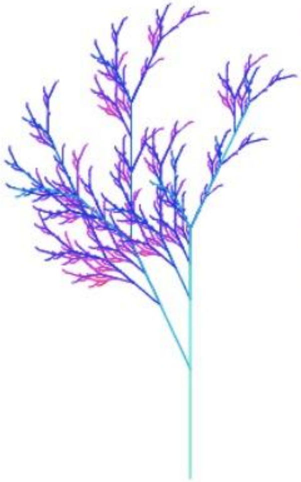
# Computer-assisted surgery

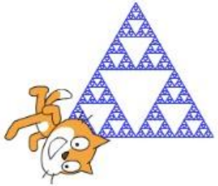
*Slide courtesy of Chris Piech; Image (c) 2012 Intuitive Surgical, Inc.*

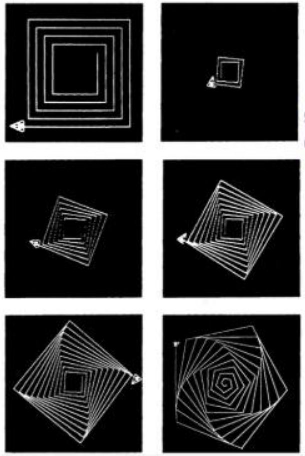
# Self-driving cars

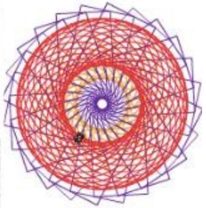
Image (c) 2019 The Drive Media, Inc. [source]

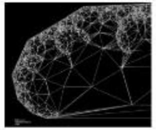
# Art

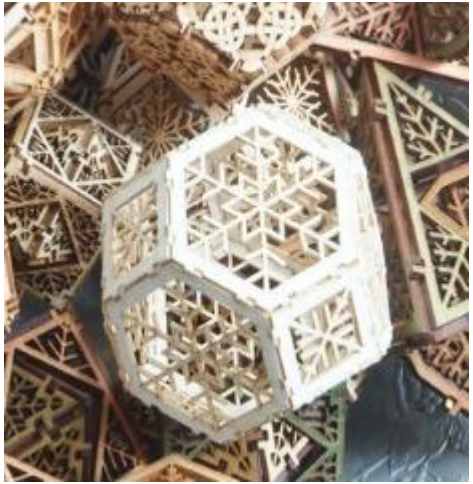


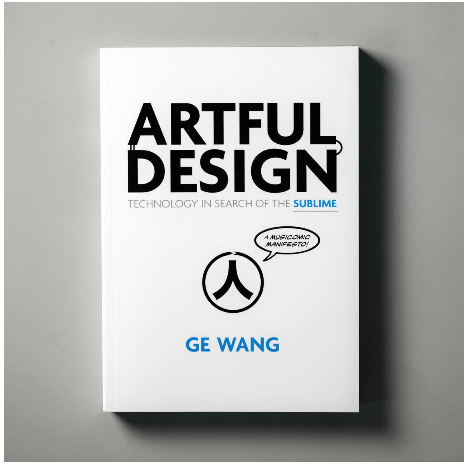




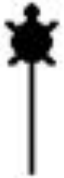




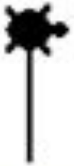




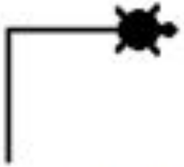
*Slide courtesy of Jenny Han*



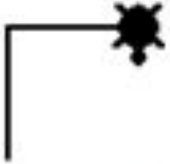
forward 50



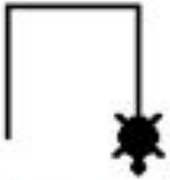
right 90



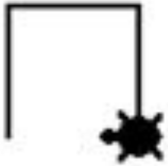
forward 50



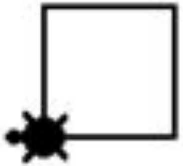
right 90



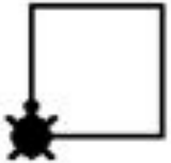
forward 50



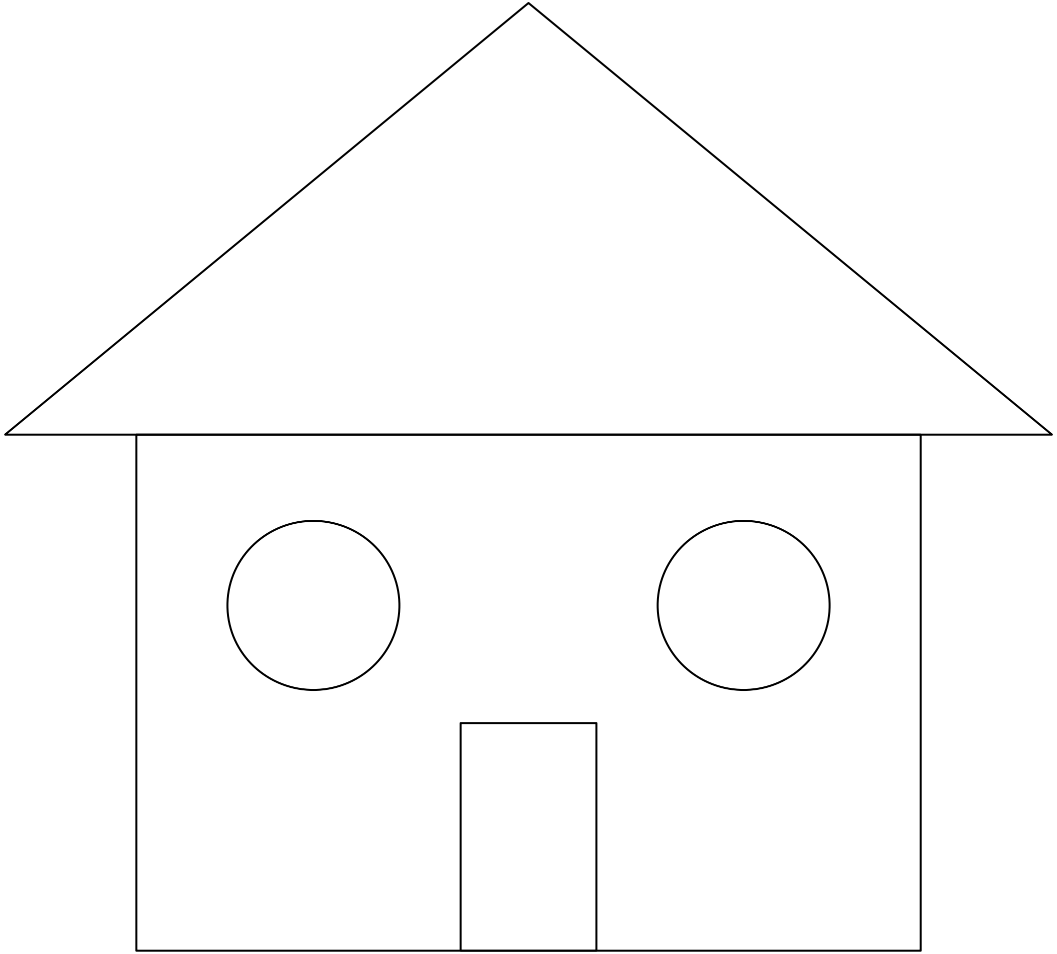
right 90

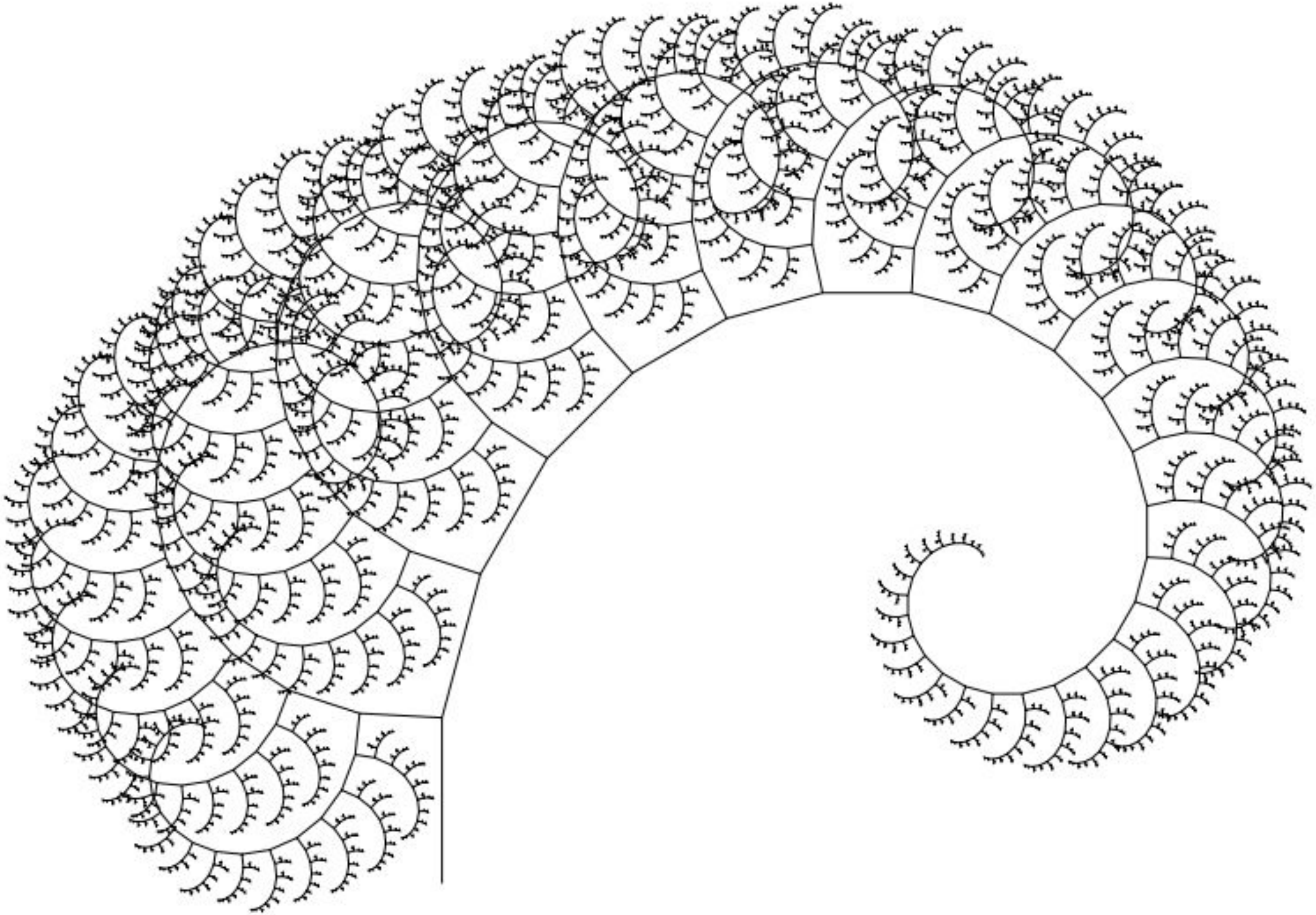


forward 50



right 90





# CS as self-expression