

Code in Place



CS49

MATTHEW RASCOFF
VICE PROVOST FOR DIGITAL EDUCATION
STANFORD UNIVERSITY

SARAH KHAN
CODE IN PLACE SECTION LEAD
FOOTHILL COLLEGE

AGENDA



1

**What is Code
in Place?**

2

**Code in
Place Comes
to Foothill
College**

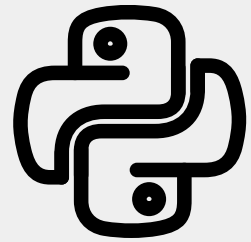
3

**Section
Leaders**

4

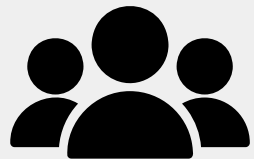
**Section is in
Session!**

CiP OVERVIEW



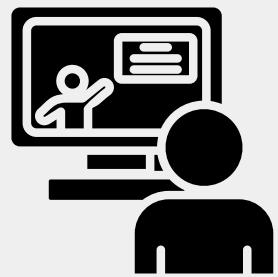
Remote Python Programming

Stanford's Code in Place is a free online course offering an introduction to Python programming, covering the first half of Stanford's CS106A.



Scalable Education Model

With over 30,000 students globally since 2020, the program combines recorded lectures and guided learning sections.



Human-Centered Approach

Sections led by volunteer instructors ensure personalized guidance and enhanced engagement.

Code in Place

A free, human-centric, intro-to-coding course from Stanford University

[Learn to Code](#)

[About Code in Place](#)

[SL App](#)



PAST OFFERINGS

3,000

section leaders teach

30,000

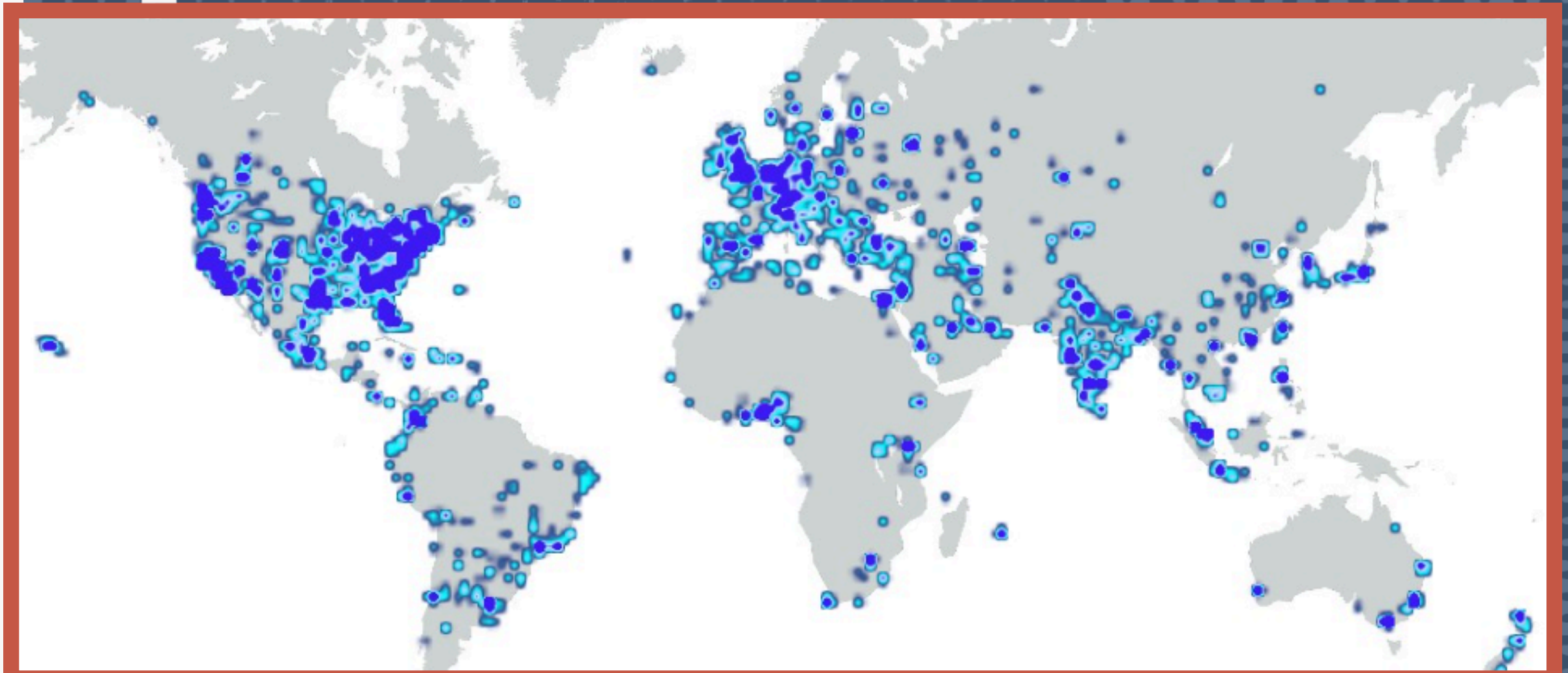
students

1:10

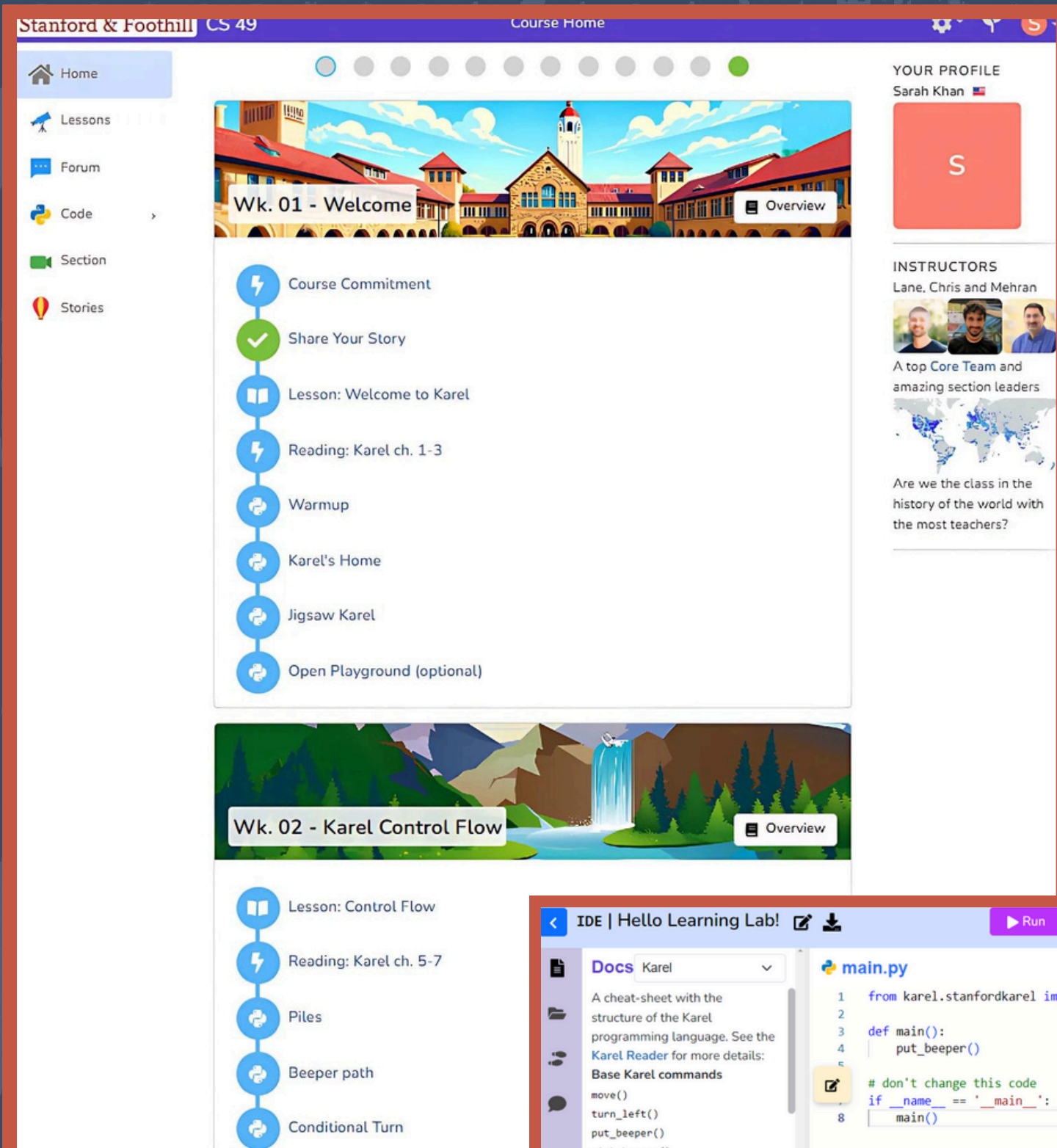
student to section leader ratio

1/2

of Stanford's CS106A



Map of students from Code in Place 2020, the first two offerings. The course had 10,000 students in the first offering and 12,000 in the second

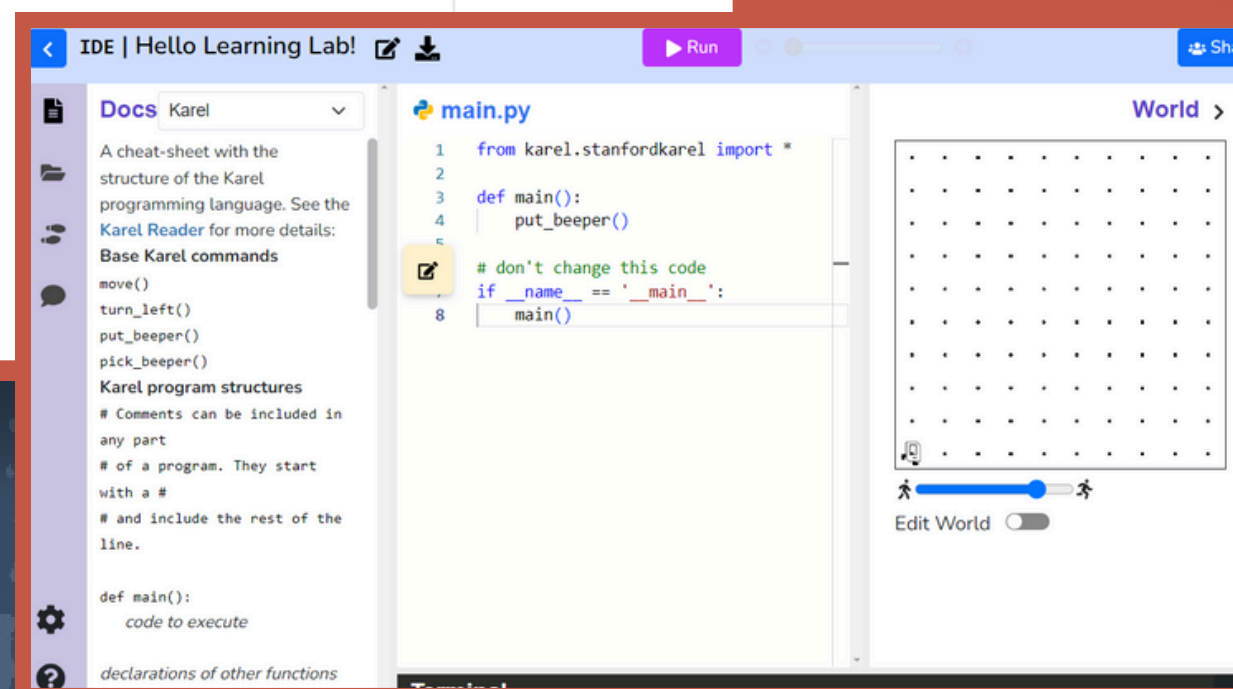


Feature Rich

The platform provides ample modularity with fast tailored feature releases.

Easy to Navigate

User Interface provides clear curriculum sequence and easy access to core course components.



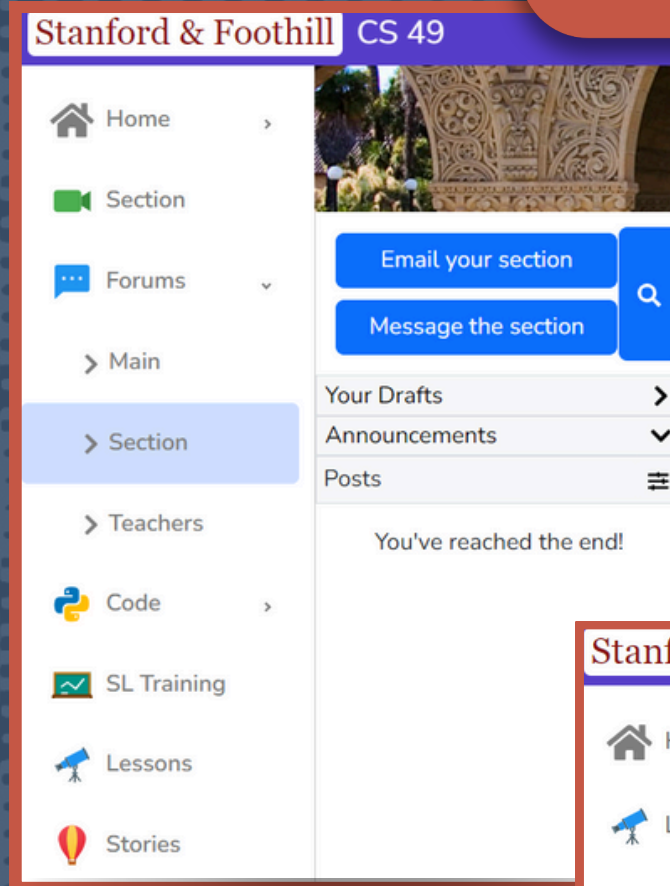
Embedded Coding Environment

Students quickly begin coding without the pain point of environment and language installation.

PLATFORM

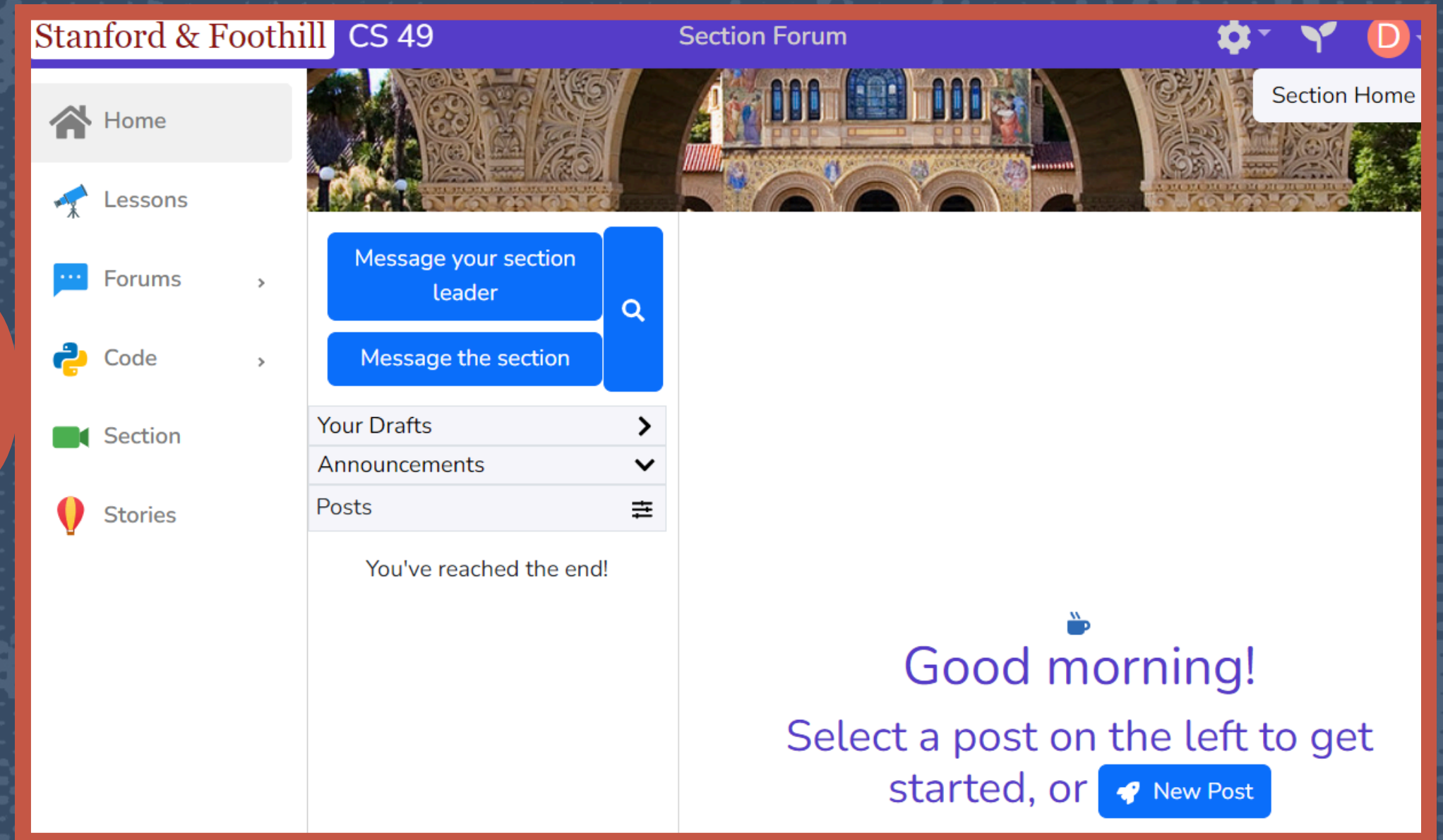
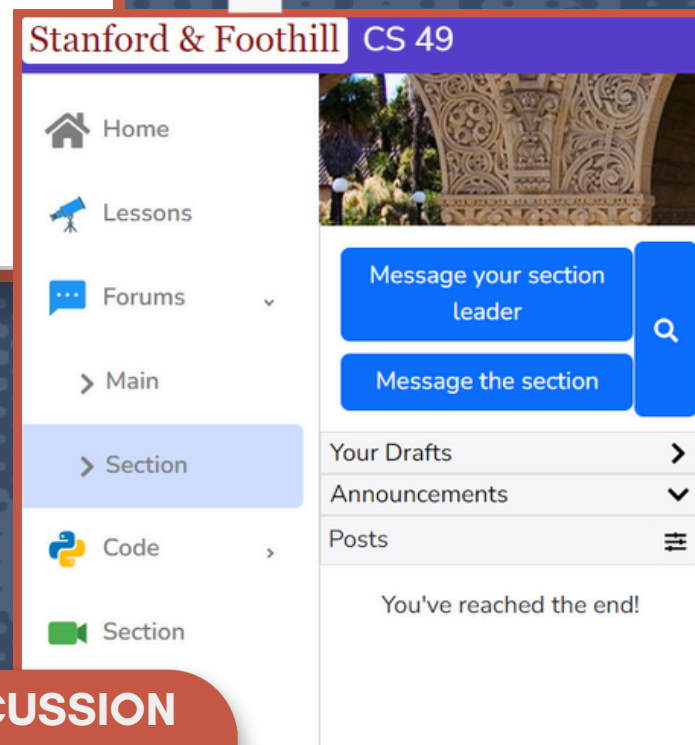
COMMUNITY

SL DISCUSSION VIEW



AI help bots are also in the works for support when teaching staff is unavailable!

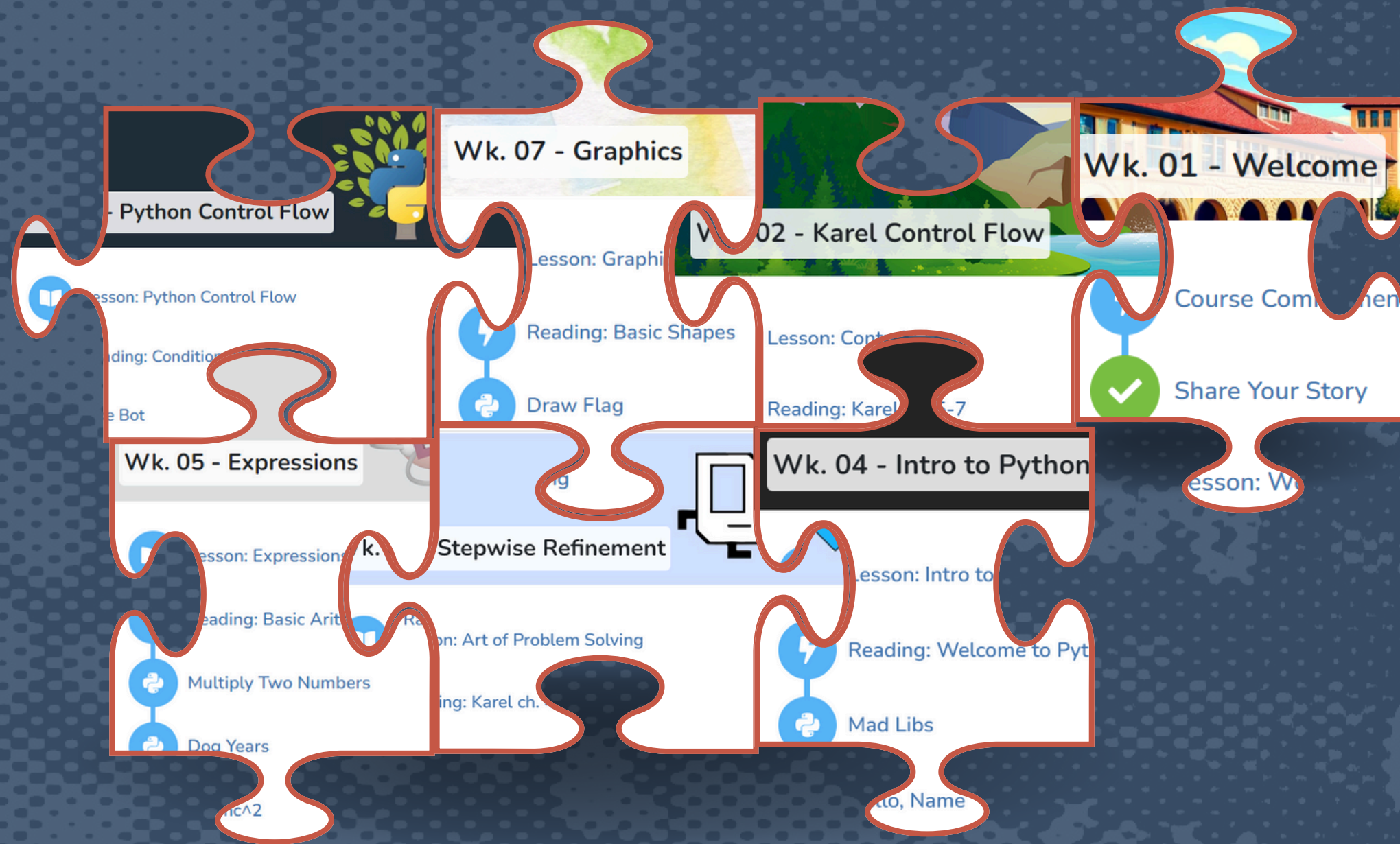
STUDENT DISCUSSION VIEW



Discussion Rich

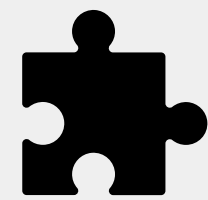
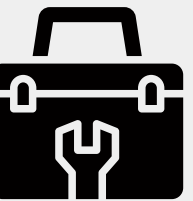
Discussion forums facilitate teaching team engagement among each other and with students. Section leads share resources and ongoing collaboration.

MODULARITY



Out of the Box

The standard Code in Place is also ready to use right out of the box, covering the first half of Stanford's CS106A.




Flexibility

Curriculum modules are customizable and easily integrate existing curriculum and course materials.



README GPL-3.0 license



Code in Place 2023 and Karel Solutions

Code in Place is an act of community service over 900 teachers from ar offer a first-of-its-kind volunteer-led course called Code in Place, t Code in Place was a great, uplifting, learning experience and over 10, in Python.

README

- ✓ If you think this repository has helped you learn something new yo
- ✗ If not, point out 'why' and spam the issue section

Contributing

Any kind of contributions to Trilochna/Code-In-Place-By-Stanford-University are welcome. While creating an issue(for this project) use Request-Feature Label.

1. [Fork](#) the Project
2. Commit your Changes
3. Open a [Pull Request](#)

Connect on LinkedIn - [@trilochnavaid](#)

Follow on Twitter - [@Trilochna_](#)

Product Solutions Resources Open Source Enterprise Pricing Sign in Sign up

Trilochna / Code-In-Place-By-Stanford-University Public Notifications Fork 1 Star 6

Code Issues Pull requests Actions Projects Security Insights

main 1 Branch 0 Tags Go to file Code

Trilochna Update README.md d188a22 · last year 174 Commits

2023 Karel	Update README.md	last year
Add Many Numbers	Update README.md	last year
Averages	Update README.md	last year
Baby Snake	Update README.md	last year
Bouncing Ball	Update README.md	last year
Carbon Dating	Update README.md	last year
Centered Square	Update README.md	last year
Chess Board	Update README.md	last year
Code In Place Resources	Add files via upload	last year
Count Nums	Create README.md	last year
Dino simulator	Update README.md	last year

About

Karel Codes

- python graphics stanford-university
- standard karel kareltherobot
- karel-python code-in-place codeinplace
- code-in-place-2023
- code-in-place-2023-solutions karel-solutions

Readme GPL-3.0 license Activity 6 stars 1 watching 1 fork Report repository

Releases

No releases published



OPEN SOURCE

GitHub Repo
 Out of the Box Code in Place is available for free via GitHub. Clone it and play around!



REMOTE + COLLABORATION



Content

CiP provides complete lessons with video lectures paired with course textbook all in one place.

Delivery Mechanism

Course materials are self contained within the platform serving as a self-contained learning experience.



Anatomy of a Program

Import Packages
main function
helper functions
start program

Chapter 4: Decomposition

As a way of illustrating more of the power that comes with being able to define new functions, it's useful to have Karel do something a little more practical than move a beeper from one place to another. The roadways often seem to be in need of repair, and it might be fun to see if Karel can fill potholes in its abstract world. For example, imagine that Karel is standing on the "road" shown in the left-hand figure, one corner to the left of a pothole in the road. Karel's job is to fill the hole with a beeper and proceed to the next corner. The diagram on the right illustrates how the world should look after the program execution.

Before:

After:

Stanford & Foothill CS 49 Course Home

Wk. 01 - Welcome Overview

- Course Commitment
- Share Your Story
- Lesson: Welcome to Karel
- Reading: Karel ch. 1-3

Stanford & Foothill CS 49 Your Section

Demo Section

Section Leader: Demo SL

Tuesdays, 3am

Next section: All done!

Join Section Zoom

Section Forum

Email Your Section

ANNOUNCEMENTS



Exercise Rich

Course materials are self contained within the platform serving as a self-contained learning experience.

Online Learning + Human Contact

The flexibility of a MOOC, with a human touch. Weekly sections give students learning support in small collaborative groups.



IDE | Example Run Share

Docs Karel

A cheat-sheet with the structure of the Karel programming language. See the Karel Reader for more details:

Base Karel commands

- move()
- turn_left()
- put_beeper()
- pick_beeper()

Karel program structures

Comments can be included in any

```
1 from karel.stanfordkarel import *
2
3 def main():
4     put_beeper()
5
6 # don't change this code
7 if __name__ == '__main__':
8     main()
```

World

Edit World

AGENDA



1

**What is Code
in Place?**

2

**Code in
Place Comes
to Foothill
College**

3

**Section
Leaders**

4

**Section is in
Session!**



CANVAS INTEGRATION

F24 CS F049 Foundations Of Computer Progra 01W, 01W Johnson 22336, 22558

FOOTHILL COLLEGE + Stanford



online!

Start Here

- read the [syllabus](#)
- consume [modules 0 & 1](#)

Code in Place

Home

Code in Place

Simple Syllabus

Announcements

Modules

Grades

Assignments

Office Hours

Foothill Tutoring

Library

Search

Accessibility Dashboard

Quizzes

Outcomes

Pages

Files

Rubrics

BigBlueButton

Collaborations

Discussions

Syllabus

Settings

COURSE HOMEPAGE

Stanford & Foothill CS49 Fall 2024 Home Page for Students

Student

Home

Admin

Teacher

Student

Section

Code

Forums

Events

SL Training

Lessons

Stories

Grades

Wk. 01 - Welcome Overview

Course Commitment

Share Your Story

Lesson: Welcome to Karel

Reading: Karel ch. 1-3

Warmup

Karel's Home

Jigsaw Karel

Open Playground (optional)

YOUR PROFILE Sarah Khan

INSTRUCTORS Lane, Chris and Mehran

Are we the class in the history of the world with the most teachers?

CiP Landing Page

F24 CS F049 01W, 01W Foundations ... > Code in Place

Search this course

Home

Code in Place

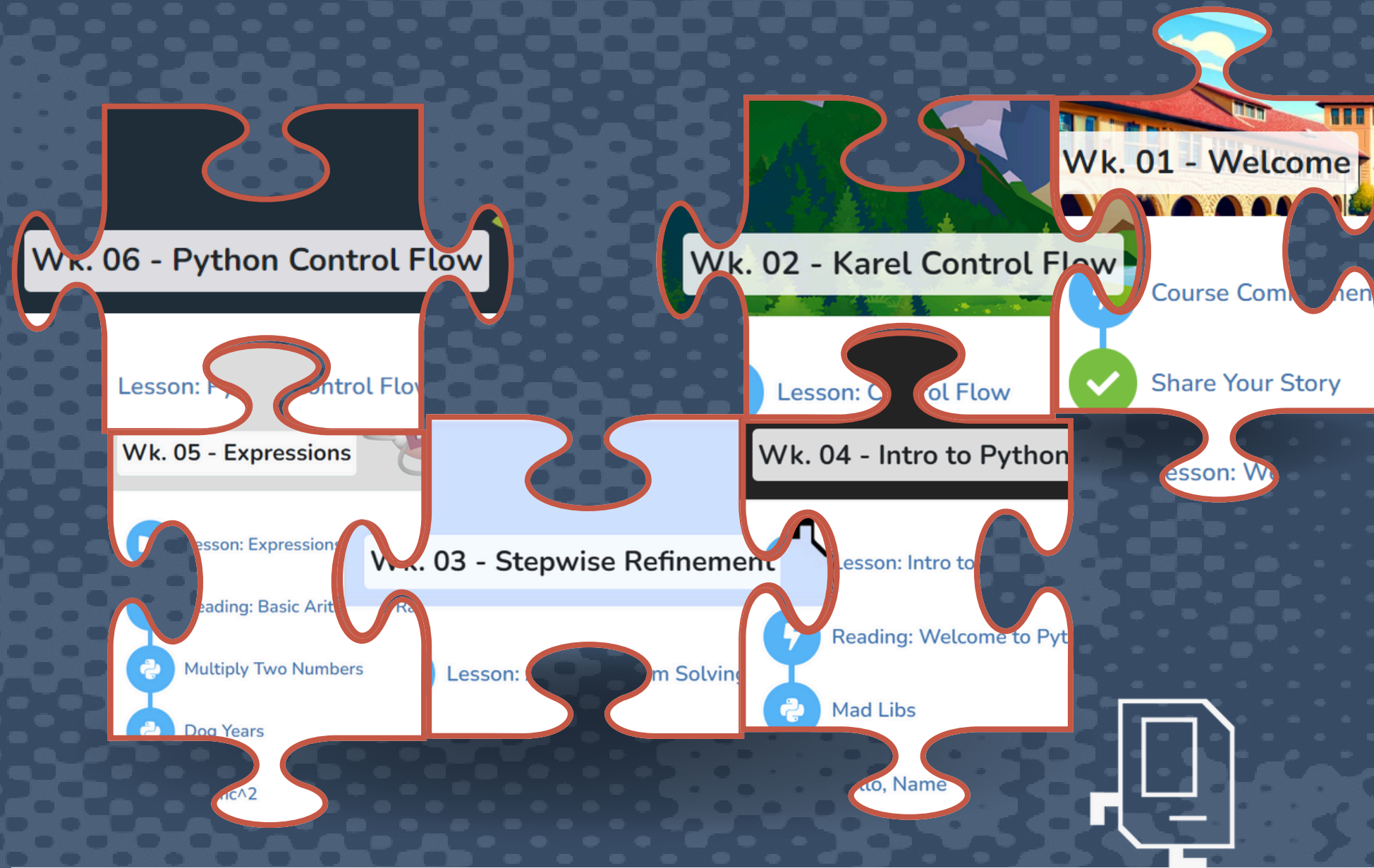
Simple Syllabus

To visit the URL, click the button below.

Open in New Tab

CiP Navigation

CUSTOMIZED CURRICULUM



Integration

Original CiP curriculum consisted of 6 modules (1 module / week). At Foothill that was extended to 11 modules using existing curriculum and all original content.

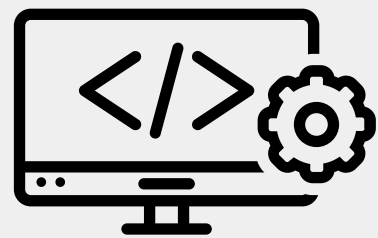
CURRICULUM





CiP for Credit

For-credit CiP is still in its early days, but there has been ample support and ongoing collaboration making it an ever growing course.



Innovation

"It's really fun to see into the development side of things and watch visions and aspirations come to life, and especially with such a shared vision of serving our students in mind."

TRAILBLAZING



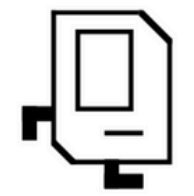
Content Integration

CiP's existing content provides a solid, well-rounded foundation that's ready to expand seamlessly to meet any curriculum requirements.

[Karel](#)

Chapter 1: Introducing Karel the Robot

In the 1970s, a Stanford graduate student named Rich Pattis decided that it would be easier to teach the fundamentals of programming if students could somehow learn the basic ideas in a simple environment free from the complexities that characterize most programming languages. Rich designed an introductory programming environment in which students teach a robot to solve simple problems. That robot was named Karel, after the Czech playwright *Karel Čapek*, whose 1923 play *R.U.R. (Rossum's Universal Robots)* gave the word robot to the English language.

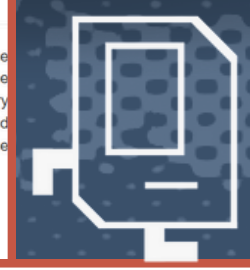


Karel the Robot was quite a success. Karel has been used in introductory courses around the world and has been taught to millions of students. Many generations of programmers have worked with Karel, and it is still the gentle introduction to coding that many students remember.

What is Karel?

Karel is a very simple robot living in a very simple world. By giving Karel a set of simple commands, you can make it perform certain tasks within its world. The process of specifying those commands is called programming. Initially, Karel understands only a very small number of predefined commands.

- 1 - Meet Karel
- 2 - Programming
- 3 - New Functions
- 4 - Decomposition
- 5 - For Loops
- 6 - While Loops
- 7 - Conditionals
- 8 - Refinement
- 9 - Extra Features
- 10 - Reference
- 11 - Code



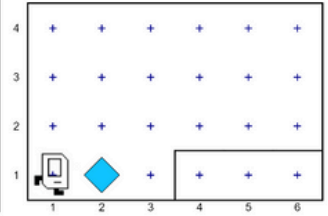
[Karel](#)

Chapter 2: Programming Karel

The simplest style of Karel program uses text to specify a sequence of built-in commands that should be executed when the program is run. Consider the simple Karel program below. The text on the left is the program. The state of Karel's world is shown on the right:

```
# File: FirstKarel.py
# -----
# The FirstKarel program defines a "main"
# function with three commands. These commands cause
# Karel to move forward one block, pick up a beeper
# and then move ahead to the next corner.
from karel.stanfordkarel import *

def main():
    move()
    pick_beeper()
    move()
```



▶ Run Program Show Text Descriptions

Press the "Run" button to execute the program. Programs are typically written in a special application called an **Integrated Development Environment (IDE)** and most Karel programs are written in an IDE called PyCharm. Like an IDE, this reader has the ability to execute programs in order to help you see how things work as you learn.

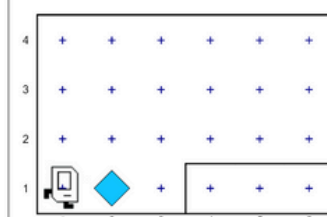
[Karel](#)

Chapter 3: Defining New Functions

In the last chapter we wrote a program to help Karel climb a simple ledge:

Example: FirstKarel

```
# File: FirstKarel.py
# -----
# Karel picks up a beeper and places it on a ledge.
from karel.stanfordkarel import *
def main():
    move()
    pick_beeper()
    move()
    turn_left()
    move()
    turn_left()
    turn_left()
    turn_left()
    move()
    put_beeper()
    move()
```



▶ Run Program Show Text Descriptions

Even though the `FirstKarel` program above demonstrates that it is possible to perform the `turn_right()` operation using only Karel's built-in commands, the resulting program is not particularly clear conceptually. In your mental design of the program, Karel turns right when it reaches the top of the ledge. The fact that you have to use three `turn_left()` commands to do so is annoying. It would be much simpler if you could simply say `turn_right()` and have Karel understand this command. The resulting program would not only be shorter and easier to write, but also significantly easier to read.

CONTENT



AGENDA



1

**What is Code
in Place?**

2

**Code in
Place Comes
to Foothill
College**

3

**Section
Leaders**

4

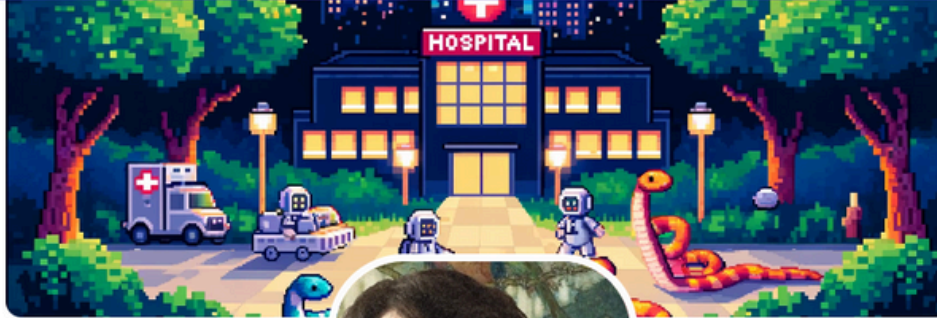
**Section is in
Session!**


WHAT ARE SECTION LEADERS?



Stanford & Foothill CS 49 Your Section

Home Section Forums Code SL Training Lessons Stories

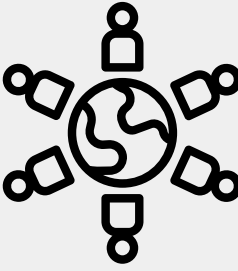



Sarah's Section
Section Leader: Sarah K
Thursdays, 8pm
Next section: All done!

[Join Section Zoom](#)
[Section Forum](#)



Diversity
Section leaders come from diverse backgrounds with a shared mentality centered on helping guide students into the world of programming.



Stanford & Foothill CS 49 Your Section

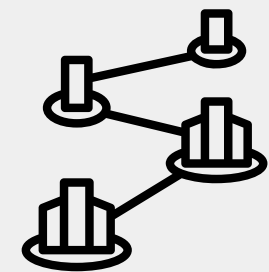
Home Teacher Student Section Forums Code SL Training Lessons Stories

SECTION SCHEDULE

#	Date	Topic	Docs	Atten.
1	April 16th, 3:00 AM	Karel Control Flow		✗
2	April 23rd, 3:00 AM	Stepwise Refinement		✗
3	April 30th, 3:00 AM	Intro to Python		✗
4	May 7th, 3:00 AM	Expressions		✗
5	May 14th, 3:00 AM	Python Control Flow		✗
6	May 21st, 3:00 AM	Graphics		✗
7	May 28th, 3:00 AM	Functions		✗
8	June 4th, 3:00 AM	Animation		✗
9	June 11th, 3:00 AM	Lists		✗
10	June 18th, 3:00 AM	Dictionaries		✗

Ever Expanding

CiP built from the ground up with 5 section leaders at the start, to 3000 section leaders now.



COLLABORATION



Training + Support + Learning Model

Section leaders meet weekly to discuss that week's modules with teaching team. CiP platform has a dedicated discussion area for section leaders and teachers as well.

Stanford & Foothill CS49 Fall 2024 Your Section

Home Admin Teacher Student Section Code Events SL Training Lessons Stories Grades Forums

Sarah's Section
 Section Leader: Sarah Khan
 Thursdays, 8pm
 Next section: October 17th (5 days and in 22 hours)

Can you attend October 17th
 Join Section Zoom
 Section Forum
 Email Your Section

ANNOUNCEMENTS

SL SECTION RESOURCES

SECTION SCHEDULE

#	Date	Topic	Handout	Code	Lesson Plan	Solutions	Attendance
1	October 3rd, 8:00 PM	Karel Control Flow					✗
2	October 10th, 8:00 PM	Stepwise Refinement					✗
3	October 17th, 8:00 PM	Intro to Python					
4	October 24th, 8:00 PM	Expressions					

Stanford & Foothill CS49 Fall 2024 Your Section

Home Section Code Events SL Training Lessons Stories Grades Forums

Week 2 section: Karel control flow

This week in section, your first priority is to meet your section leader and discover what sections in Code in Place are all about. Your section leader will therefore spend the first part of section on introductory logistics. Afterwards, you'll solve a Karel problem together using decomposition and stepwise refinement.

1. Hospital Karel

Countries around the world are dispatching hospital-building robots to make sure anyone who gets sick is treated. They have decided to enlist Karel robots, and your job is to program those robots.

Karel begins at the left (west) end of a row that might look like this:

HANDOUT



CODE

IDE | Hospital Karel

Instructions: Your country is prototyping hospital-building robots. They have decided to enlist Karel robots. Your job is to program those robots. Karel begins at the left end of a row that might look like this:

```

main.py
1 from karel.stanfordkarel import *
2
3 # Here is a place to program your Section problem
4
5 def main():
6     """
7     You should write your code to make Karel do its task in
8     this function. Make sure to delete the 'pass' line before
9     starting to write your own code. You should also delete the
10    comment and replace it with a better, more descriptive one.
11    """
12    pass
13
14 if __name__ == "__main__":

```

World: Hospital 1

Stanford & Foothill CS49 Fall 2024 Your Section

Home Section Code Events SL Training Lessons Stories Grades Forums

Week 2 section: Karel control flow (Solution)

Hospital Karel

```

Program: Hospital Karel
Karel traverses 1st street from west to east, building hospitals
wherever it encounters a beeper.
"""
from karel.stanfordkarel import *
"""
Program: Hospital Karel
Karel traverses 1st street from west to east, building hospitals
wherever it encounters a beeper.
"""
def main():
    while front_is_clear():
        if beepers_present():
            build_hospital()
        safe_move()

```

SOLUTIONS

Stanford & Foothill CS49 Fall 2024 Your Section

Home Section Code Events SL Training Lessons Stories Grades Forums

Week 2 Section Lesson Plan

This is going to be the first week of coding for your students. What a beautiful thing :-).

This lesson plan is a guide you can use to teach section this week. If you would like, we have also linked a few videos of past SLs teaching a demo section or a recorded section. You can check these out at the bottom of the handout.

Sample Section Teaching Videos

- One of the Head TAs giving a demo teach

Helpful Tools

- <https://karelhelper.com/>
- Ice Breaker Ideas
- Slide Deck from a Past SL
- Slide Deck from the Head TAs

Hospital building Karel

Here is the section handout the students will see Here are the section solutions you can use as a reference. Students will see these one hour after their section starts.

LESSON PLAN

SL CONTENT

FLEXIBILITY

CODE IN PLACE WEEK 2

Welcome to Section!
I'm Sarah, your Section Leader

DECOMPOSITION

Purpose: To break complex sequences into digestible "batches" of steps

Program Outcome: What is the program's objective?

Functions: The ingredients of each program

Steps: Individual steps in each function that make up the individual ingredients.

Helper Functions:

- Functions that support other functions
- Good for reuse

Recipe

INGREDIENTS

DIRECTIONS



CONTROL FLOW

FOR-LOOP	WHILE-LOOP	IF STATEMENTS
Syntax: for 1 in range(#): (Do this) step 1 step 2	Syntax: while (active condition)(): (Do this) step 1 step 2	Syntax: if (active condition is true): (Do this) step 1 step 2
What are they? Run through a set of steps a defined number of times.	What are they? Continuously cycles through defined step while the condition is active. Inactive condition means it's invalid and loop stops. !! CAUTION !! While loops are often the source of infinite loops!	What are they? Steps that are only activated when a condition is met.

SECTION PROBLEM

HOSPITAL KAREL

Let's build these hospitals!

Grid-based programming challenge for 'Hospital Karel' showing a grid of dots and blue diamonds, with a small blue character icon.



Fun Environment

Section content follow CiP lesson plans, but are flexible to section leader teaching styles and provide a fun programming environment and challenges.

AGENDA



1

**What is Code
in Place?**

2

**Code in
Place Comes
to Foothill
College**

3

**Section
Leaders**

4

**Section is in
Session!**

CONTACT US

Interest Form



THANK
YOU!



Code in
 Place

MATTHEW RASCOFF

VICE PROVOST FOR DIGITAL EDUCATION
STANFORD UNIVERSITY

MIKE ACEDO

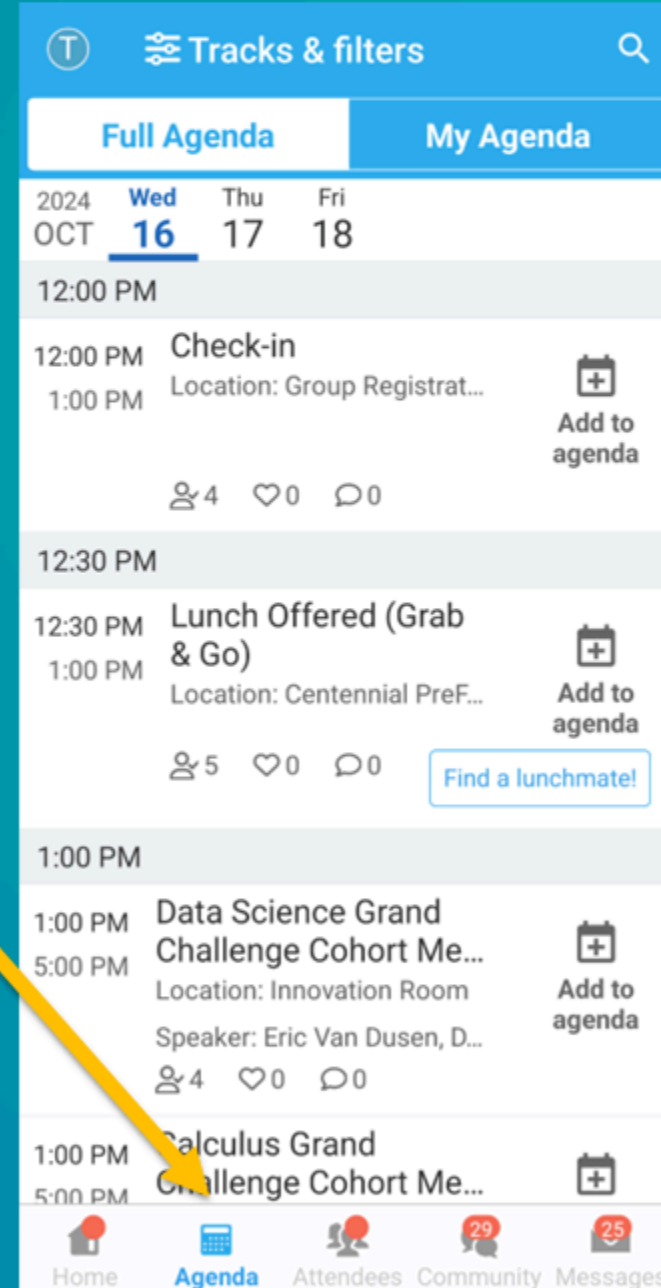
ASSISTANT DIRECTOR, PROJECT
INNOVATION & TECHNOLOGY
STANFORD UNIVERSITY

MACEDO@STANFORD.EDU

Session Feedback

We'd love to hear from you! Please let us know your thoughts on this session by filling in the feedback form in Whova. Each session feedback survey you complete in Whova will enter you into a prize drawing for a \$25 Amazon gift card (hard copy submissions are not eligible, but multiple Whova entries are allowed!). Your input is greatly appreciated!

Step 1: go to the Agenda and select the appropriate session



Step 2: select the Session Feedback button to fill out the brief survey for this session

