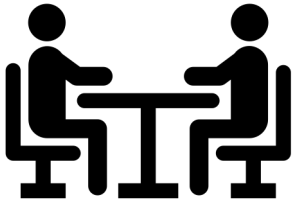
Adam's Section

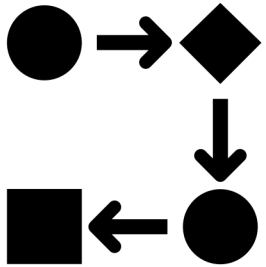
CS106A - Code in Place 2023

Welcome and congratulations!

What is an online Section?

- A more intimate, one-on-one environment for you to get a recap of this week's content, ask any questions, and work through an example together!





A little bit about me

Grew up in 🇲🇨
Originally from 🇬🇧 🇮🇹 🇨🇦

BSc
Computer Science

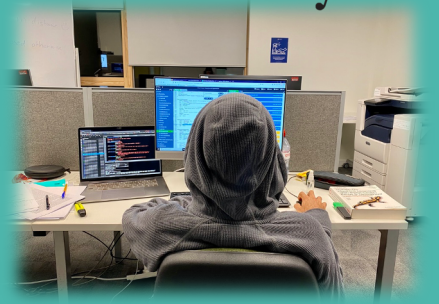
University of
Bath

Master's
Artificial Intelligence

University of
St Andrews

SCUDERIA
Toro Rosso

New Day



A little about you!

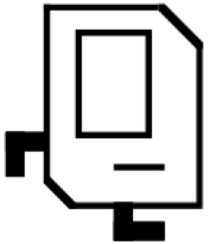
- 📣 Introduce yourself
- 🌍 Where in the world you are right now
- 💼 What you do in life
- 🤗 Why are you excited about Code in Place?
- 🤩 1 interesting fact about you!
- 🗣️ Name someone else to introduce themselves

Core Values

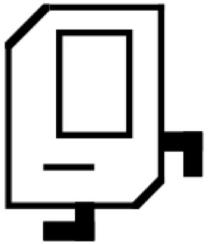
- **Interactivity and Engagement**
  - Section != lecture
  - Section is about **collaborating** on code and building a **community**.
  - There's no one right answer
  - Mistakes are a good thing!
  - Ask Ask Ask!
- **Respect**
- **Open-Mindedness and Patience**
- **FUN**

Course Logistics & Communication

- Section is the same time every Friday for 6 weeks
- I will join the sections 5 minutes before and stay 15 minutes after for any questions
- Use our Section's forum to ask me or your section members any questions (about the exercises, the lectures, or to start interesting conversations
- Please mute your mics when you join (for minimum background noise) and minimize any distractions for you (close those tabs!)
- Feel free to unmute at any time to stop me and ask anything!
- If you're comfortable, keep cameras on so we can all see each other!



Code in Place



Section 1: Welcome to Section

Week 1: April 28th, 2023

Where We Are

- Course Logistics
- Getting to know each other
- Review concepts
- Work through a problem together

Learning Goals

- How to approach a problem?
- Recapping decomposition and loops
- Defining helper functions
- If statements
- Practicing loops
  - How to decide between a while and a for loop

Approaching a problem

Understand	Strategise	Translate
• Write stuff down. • Explain problem out loud to someone. • Draw a picture.	• Walk through an example or two. • Look for patterns. • Think about what things you'll need: loops, conditions, helper functions, etc.	• First into **pseudocode**. • Then into Python.

Concept Review

- Decomposition
- While vs. For Loop
- If Statements

Decomposition

- Break a problem into smaller problems that are more manageable to solve to repetition and code getting too long/complex.
- More of an art than a science.
- Functions should be short and read like English.
- If you repeat things (or find yourself hitting copy/paste), take a step back.

Decomposition

- How:

Define your function like this

```
def function_name():
    # <write code here>
```

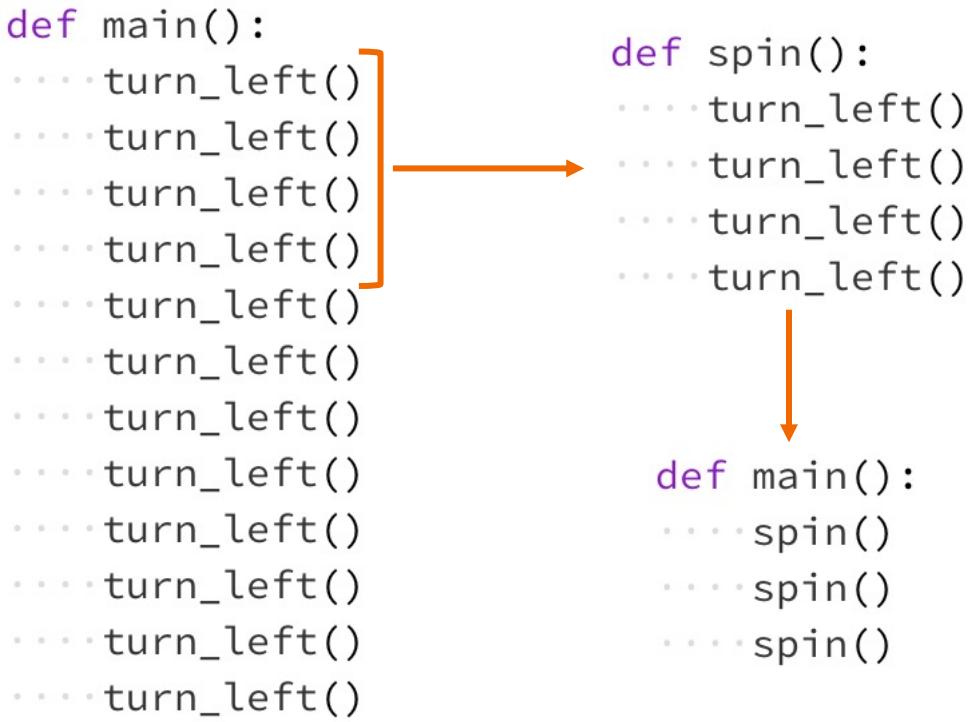
Call/use function like this

```
function_name()
```

- Pre/post conditions: each helper function defines a contract (Karel should end in the same state that it started in)

Decomposition

- Example:



Decomposition

- Key takeaway: making each function have a single responsibility

```
def spin():
    turn_left()
    turn_left()
    turn_left()
    turn_left()
```

Control flow: While vs. For Loops

- You're baking at home, and you don't know how long the cake should stay in the oven. All you know is that you should keep it in the oven until the top is golden.

While Loop

```
# Keep the cake in the oven until its top is golden
while top_is_not_golden():
    cake_in_oven()
```

- You're baking some cookies but the oven is only big enough to hold one cookie at a time. You have 12 cookies so you know that you need to use the oven 12 times.

For Loop

```
# Use the oven 12 times
for i in range(12):
    put_cookie_in_oven()
```

Control flow: While vs. For Loops

While Loop:

- When you aren't sure the exact number of times the loop should iterate
- We weren't sure how many minutes we needed before the cake was ready

For Loop:

- When you know the exact number of times the loop should iterate
- We knew we needed to put exactly 12 cookies in the oven

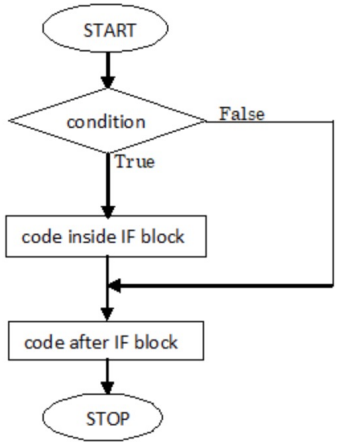
Control flow: If/Else statements

- Also called *branching*
- Making decisions based on **conditions**

If

Do something if a condition is met:

```
if front_is_clear():
    move()
```



If/Else

Also do something else when a condition is not met:

```
if front_is_clear():
    move()
else:
    turn_left()
```

Let's get some practice together!

Karel Hospital Problem

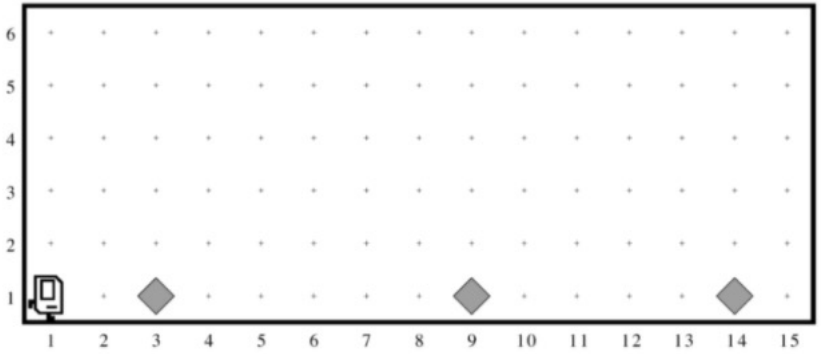
Skeleton Code

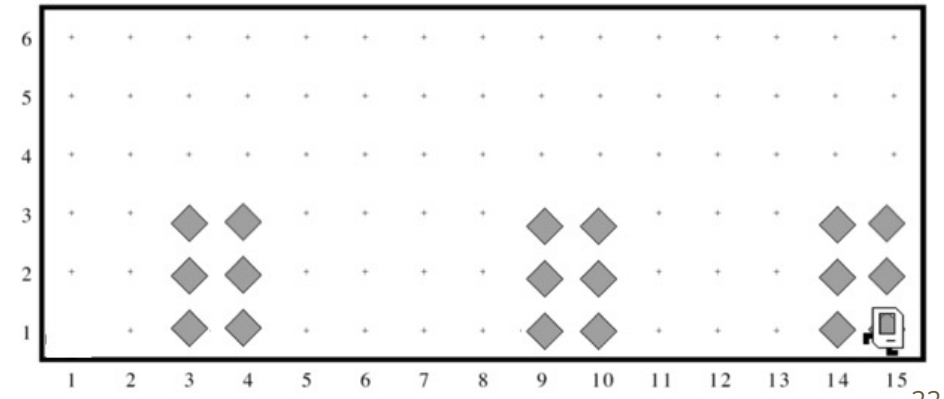
- The outer structure of the code with no functionality
- Similar across different programs

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 this
10     comment and replace it with a better, more descriptive one.
11     """
12     pass
13
14 if __name__ == '__main__':
15     main()
16
```

Hospital Karel





Wrap Up Announcements

- Section forum: Post on the forum so that everyone can learn together!
- Something I did well, something I can do better?
- Meeting next week, same time and same link!

See you next week!

Section 1: Karel the Robot