

CS Bridge, Lecture 1

Welcome to CS Bridge!



Welcome to CS Bridge!

- We are here to share with you our love for programming!
- A joint effort
- **Our goal:** form a community of people to learn and teach programming together



Bridging The World



Nice to meet you!

Instructors:



Hi! I'm Nick.



Hi! I'm Barış.



Hi! I'm Buket.

Nice to meet you!

Head Of CS Bridge:



Hi! I'm Chris.

Nice to meet you!

Head Section Leaders:



Hi! I'm Irem.



Hi! I'm Ahmet.

Nice to meet you!

Section Leaders:



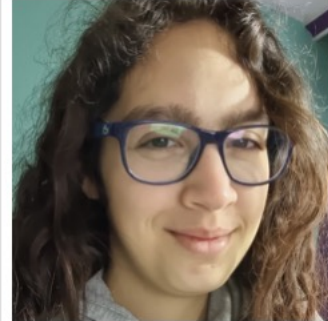
Kanoe



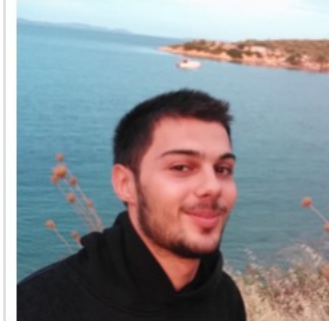
Berra



Oğuzhan



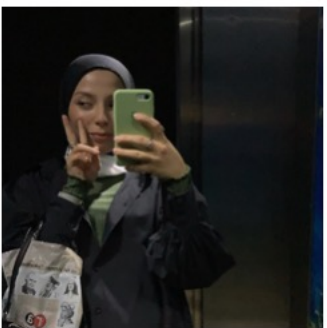
Eda



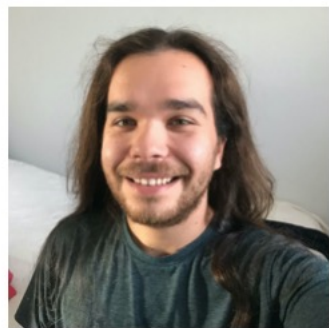
mert



Emre



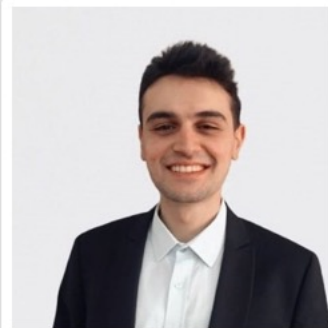
Beyza



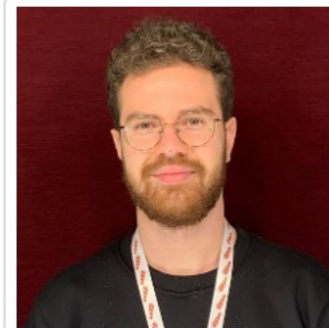
Erol



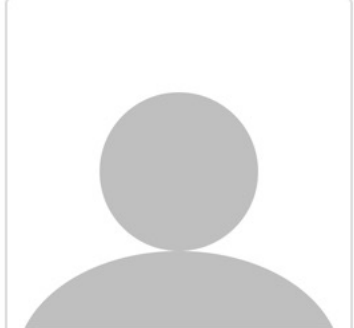
Yağız



Tarık



Ege



Tori

Nice to meet you!

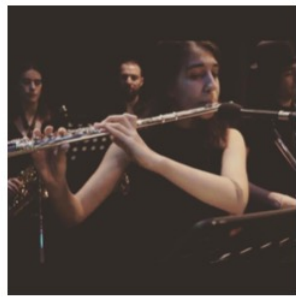
Section Leaders:



Eu Jin



Cem



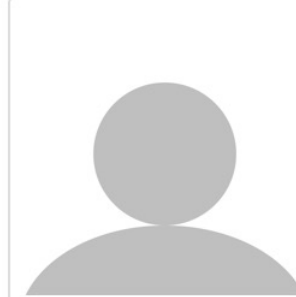
Selin



Okan



Yasin



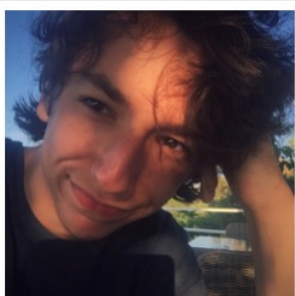
Mahmut



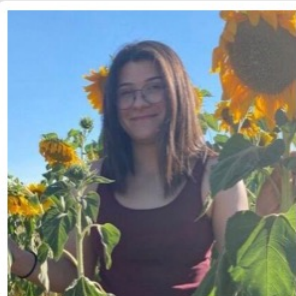
Nursena



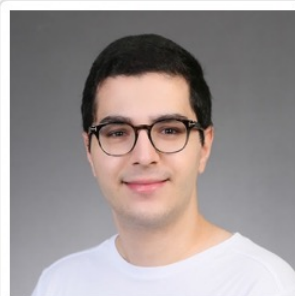
Emre



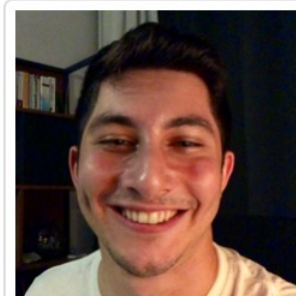
Arda



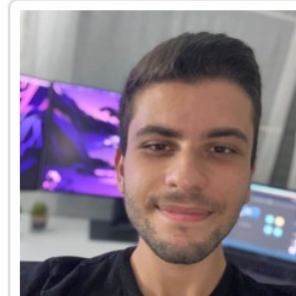
Aslihan



Can



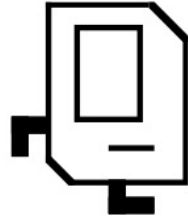
Onat



Atahan

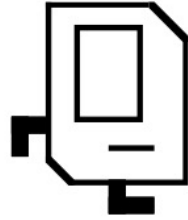
Lecture Plan

- Welcome to CS Bridge!
- Course information
- Meet Karel the Robot
- Defining new commands
- For loops



Lecture Plan


- Welcome to CS Bridge!
- **Course information**
- Meet Karel the Robot
- Defining new commands
- For loops



CS Bridge on Zoom

- You are encouraged to share video!
- Post questions/comments/follow-ups in the Zoom chat. We'll take periodic "question breaks" to address them and have teachers monitoring the chat during the lecture. Everyone is muted by default.

Course Website



turkey21.csbridge.org



The Website Tells You Where To Go!

CS Bridge

Resources ▾

Projects ▾

Examples ▾

Lectures ▾

Bonus

Sponsors



Intro to Computer Science

Summer 2021

August 2nd to August 19th, online

The program starts soon! To make sure you are ready to go, please do the following:

1. Install PyCharm and learn how to use it - information for both can be found by clicking on the "Resources" tab at the top of this page.
2. Click on the Ed account setup link you should have received in an email. Setup your account. Familiarize yourself with **Ed** and post questions with screenshots **on Ed here** if you need help with the PyCharm installation.
3. Download the "Zoom Client for Meetings" **here**.

Please read the information on the website under the resources tab, such as **General Info**, **Student Frequently Asked Questions**, and **Section Info**.

Ed Forum

- We are using a website called Ed for asking and answering questions.
- You can post questions there, and we or other students can answer them.
- If you haven't already, check your email for your invite to join Ed.
- **2 Ed Groups:**
 - **CS Bridge (main group)**
 - **Your Section**

CS Bridge Structure – Learn By Doing

- **Lecture** – Where we introduce new concepts!
- **Quickstart / Tea Time** – Right after each lecture with your section group. Time for you to play around with the material on your own computers and mingle with other students.
- **Section** – Right after evening quickstart, with your section group. You will work as a group with your Section Leader to solve practice problems.
- **Office Hours / Work** – Right after the morning quickstart and the evening section. Your own time to work on the projects and get group help from students and Section Leaders or individual help from the Section Leaders.

At the end of the morning and evening segments, submit your work!

Nooks Office Hours

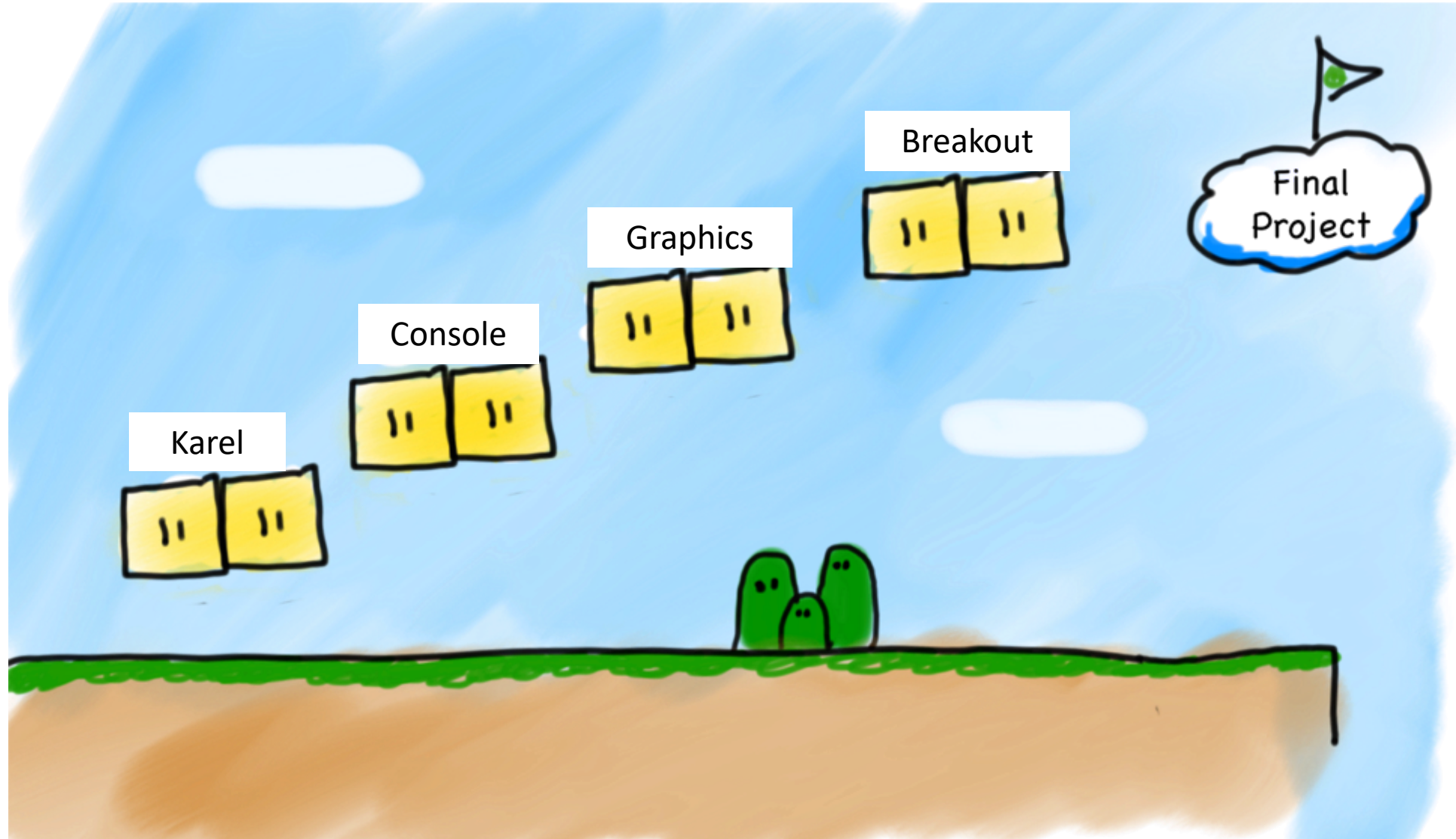
- We are using a website called Nooks for student “office hours”
- Like Zoom, but you can see all rooms, jump between rooms depending on what you are working on, and sign up in a queue for 1 on 1 help.
- You can join Nooks via a link on the course website.

Section Leader

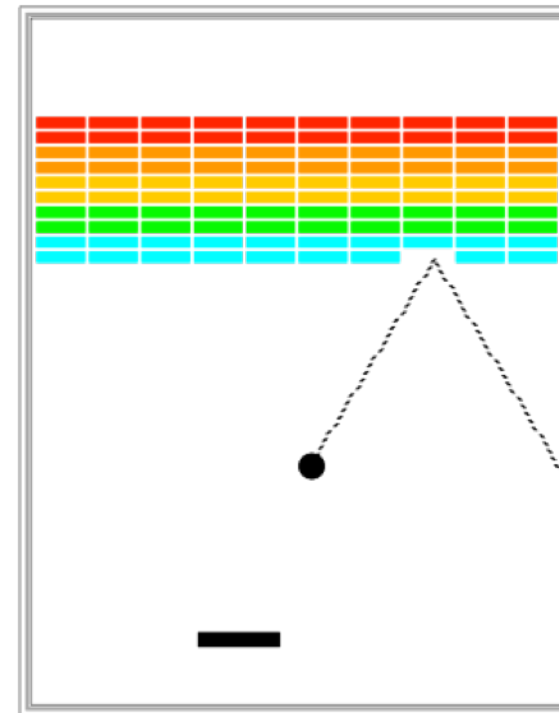
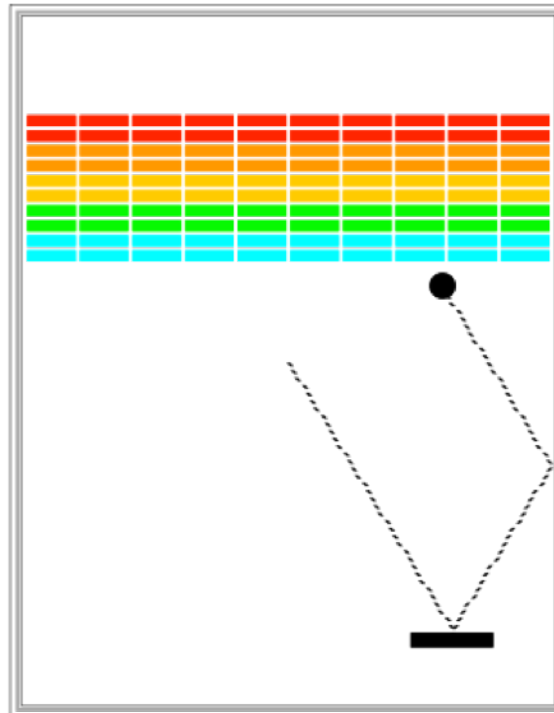
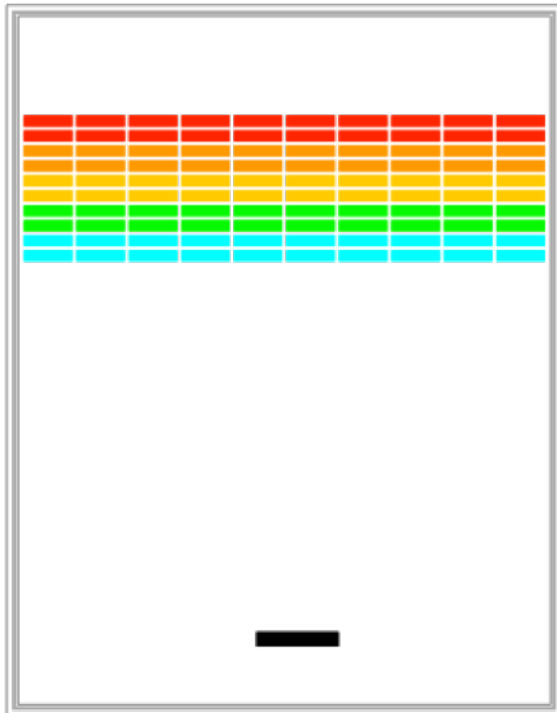
Section Leaders will...

- Lead your daily section and quickstart times
- Help you when you have questions in Office Hours and on Ed
- And much more...

What Will We Learn?



Breakout



Prerequisites

None! We assume you have never programmed before.

Art Of Computer Science

- CS Bridge covers software engineering principles – much more than just programming
- Writing a program is like writing an essay
 - Need a language (Python)
 - But just knowing the language doesn't make you a good essay writer
- Programming is a tool you can use to do amazing things!



The Joy Of Building



The Joy of



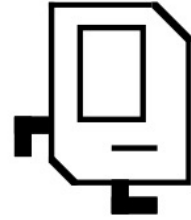
Strive For Everyone To Succeed

- We want all of you to be successful!
- You are not competing against anyone but yourself.
- Learning programming is hard! We are here to help you. Please post on Ed, come to Office Hours or talk to your Section Leader if you are having trouble.
- **Your most important task: show up!**



Lecture Plan

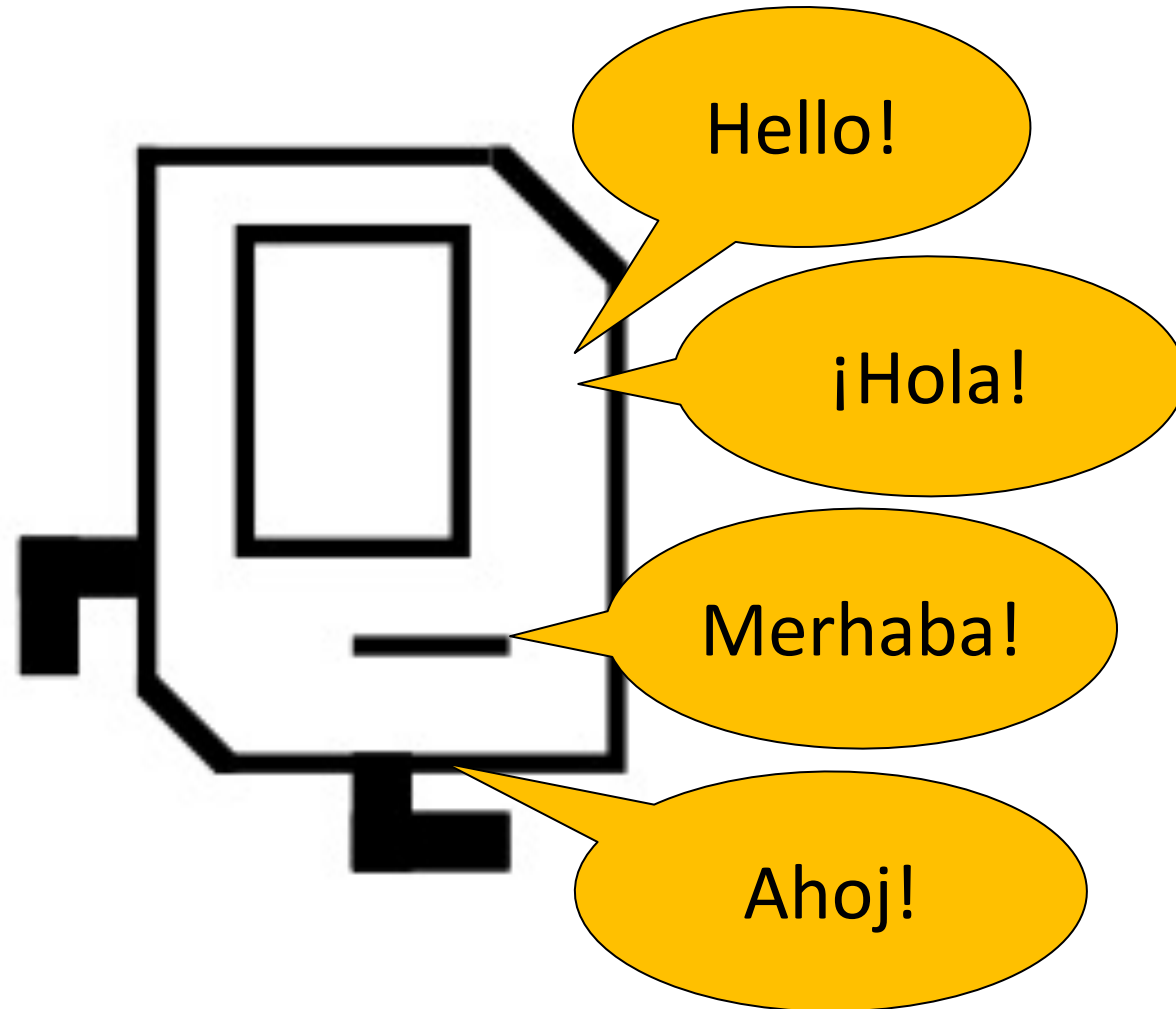
- Welcome to CS Bridge!
- Course information
- **Meet Karel the Robot**
- Defining new commands
- For loops



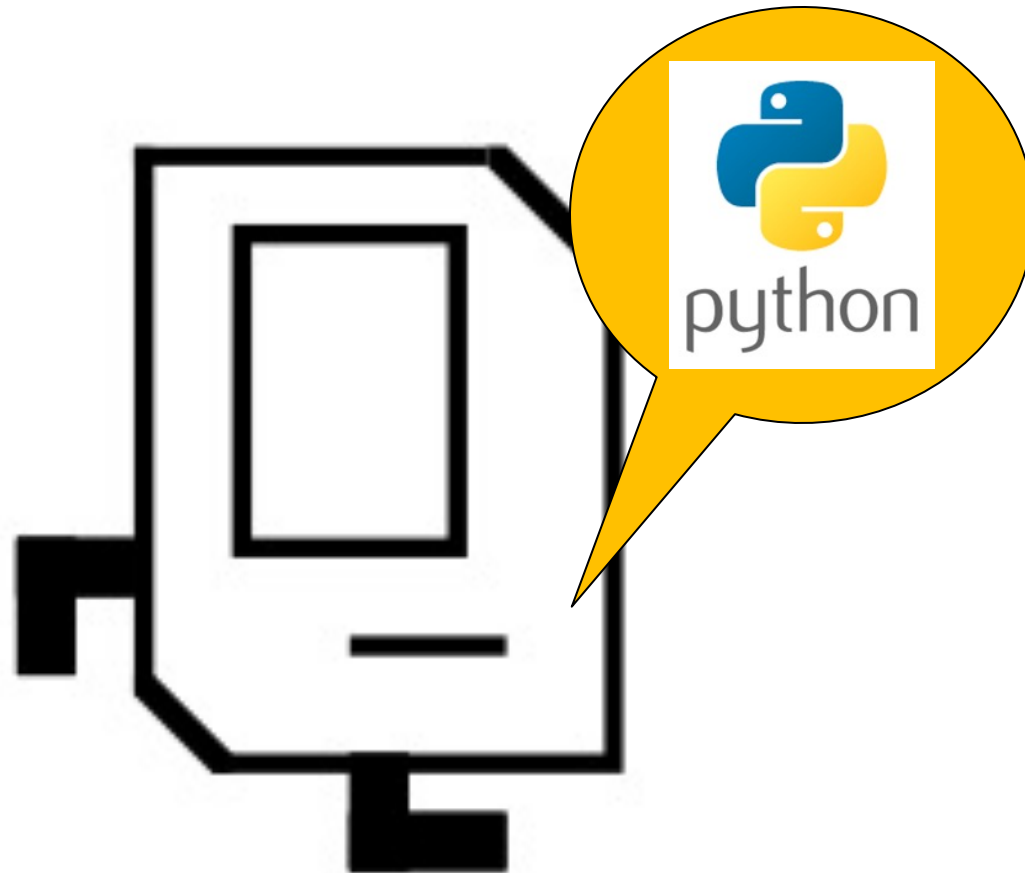
Meet Karel the Robot!



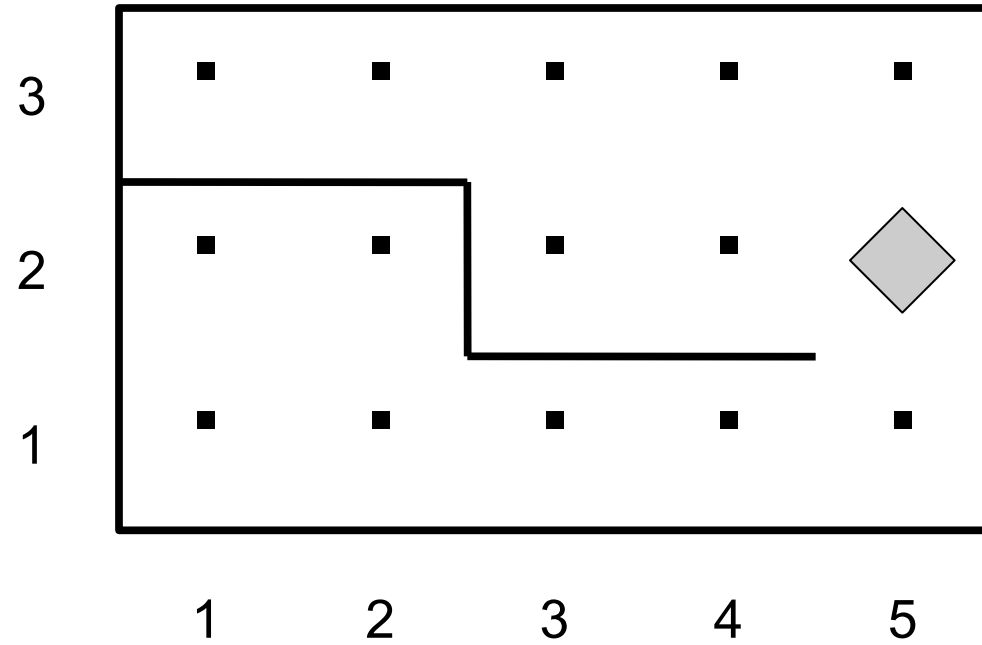
Meet Karel the Robot!



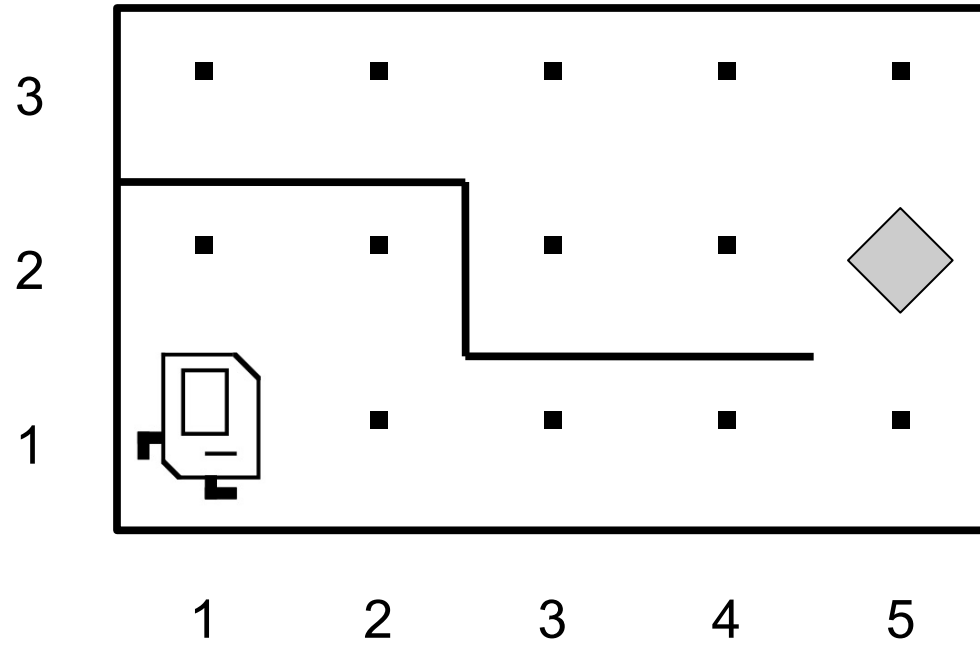
Meet Karel the Robot!



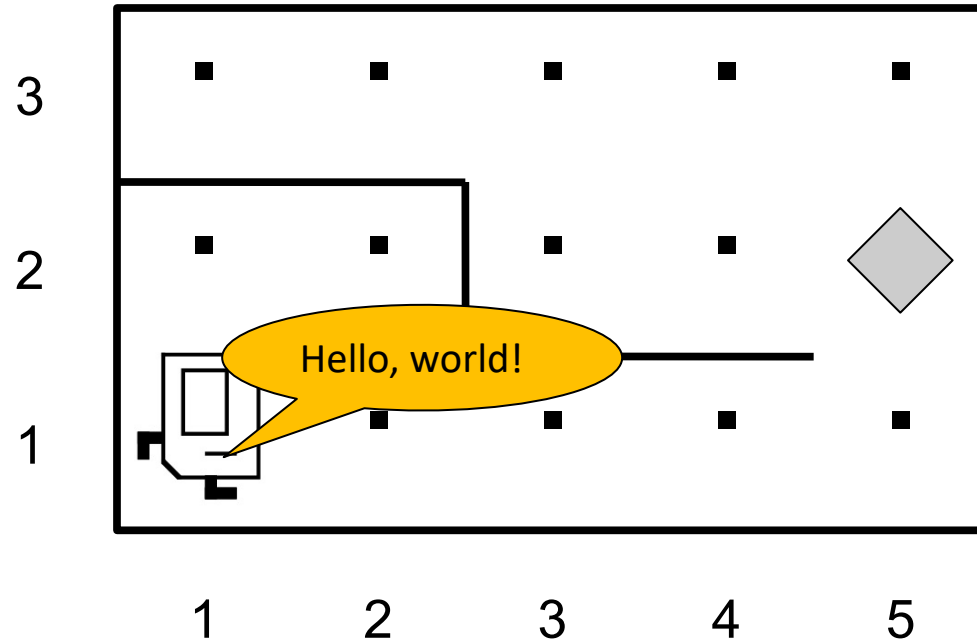
Karel's World



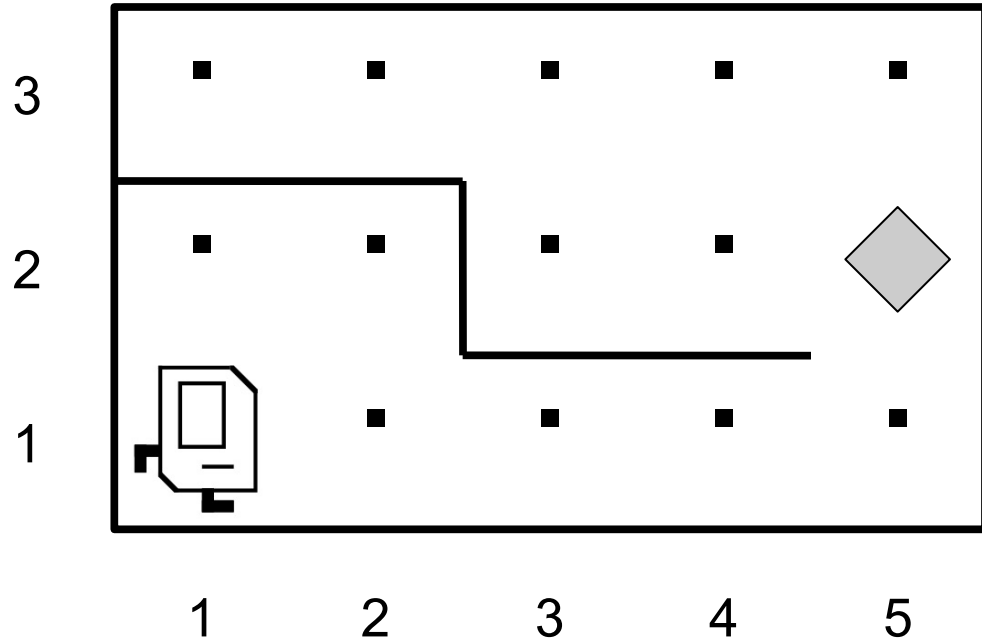
Karel's World



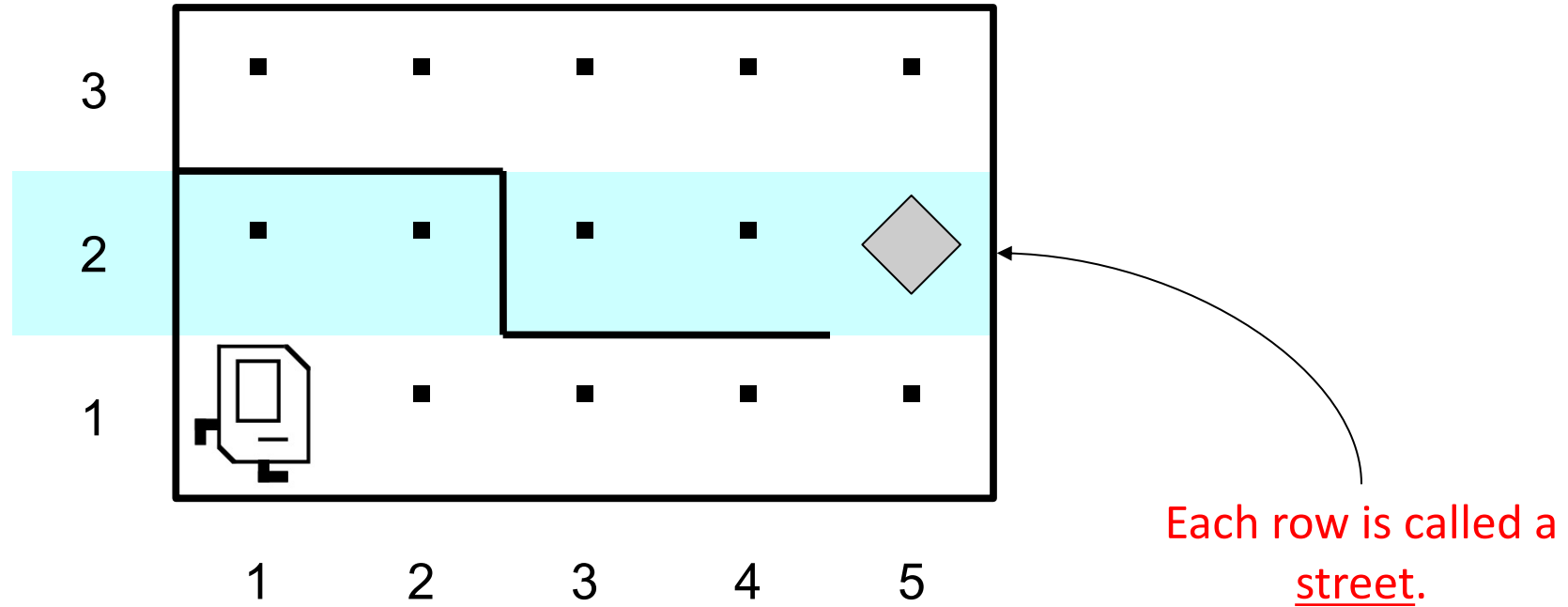
Karel's World



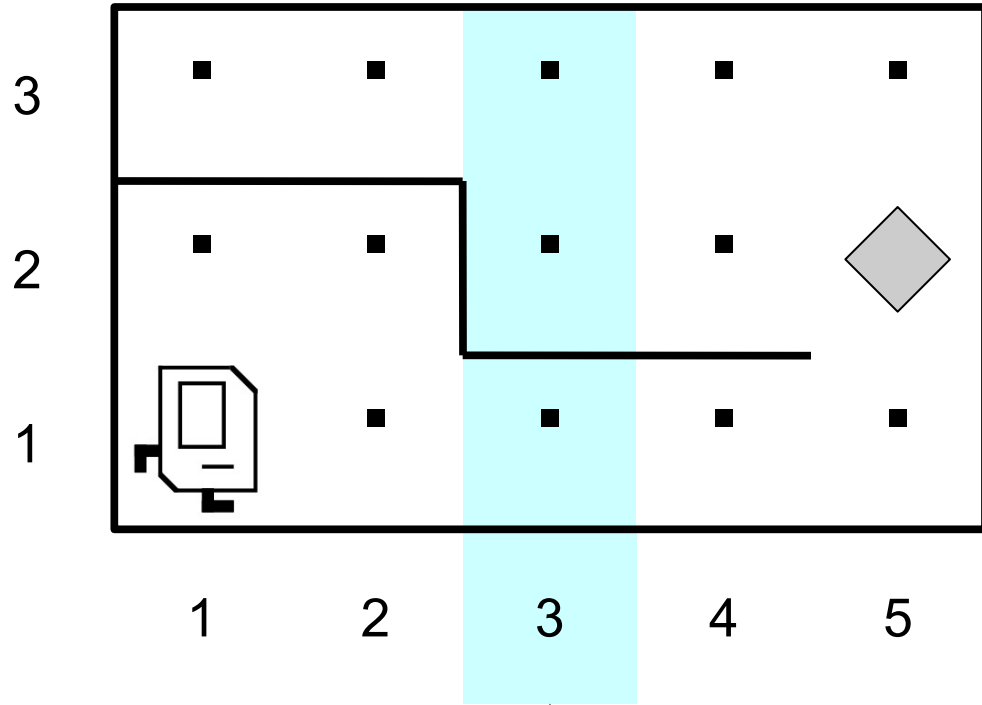
Karel's World



Streets (rows)

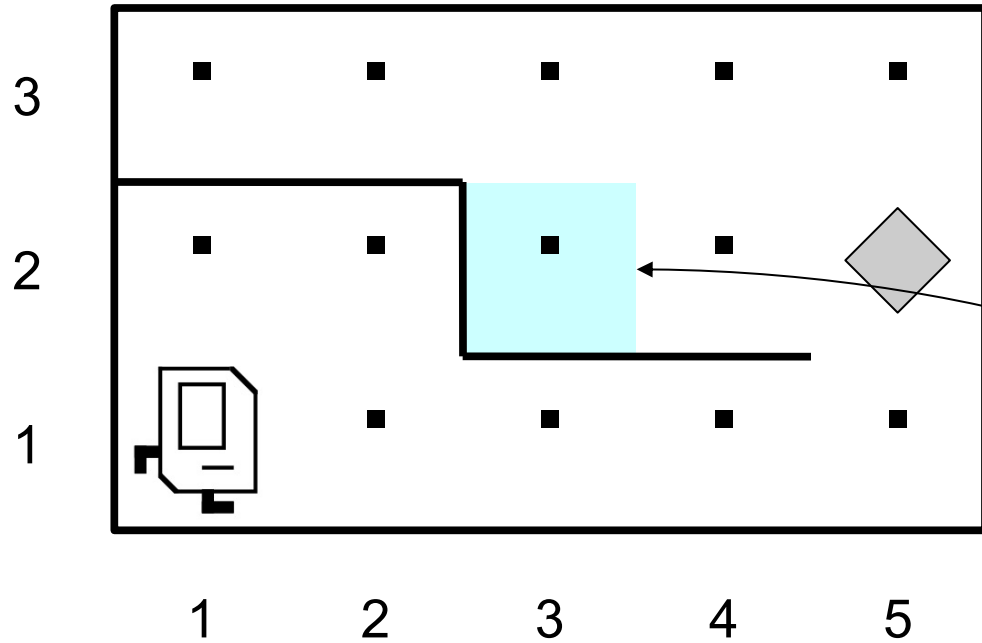


Avenues (columns)



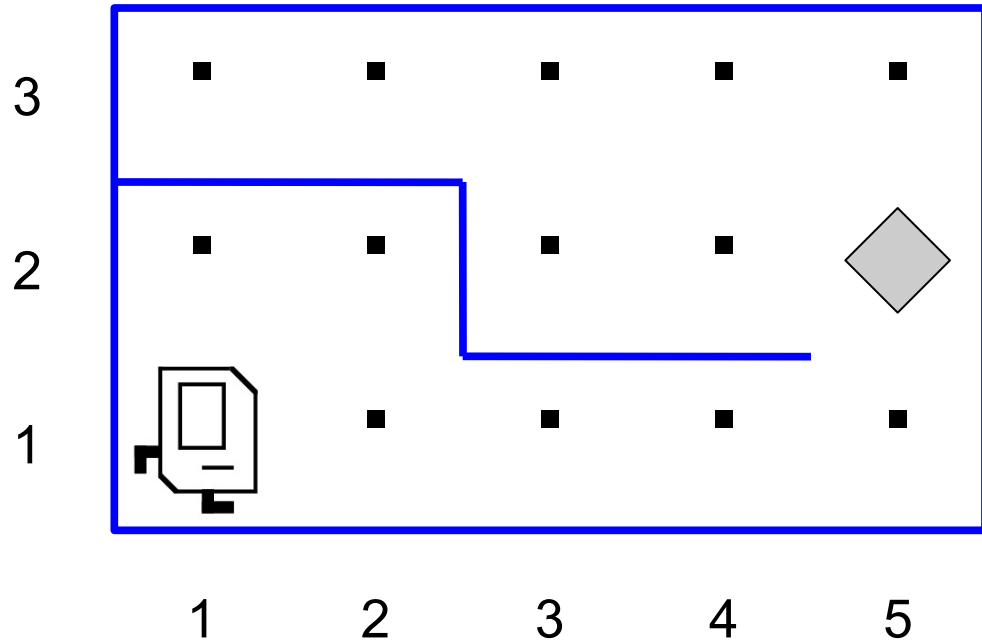
Each column is called an avenue.

Corners (locations)



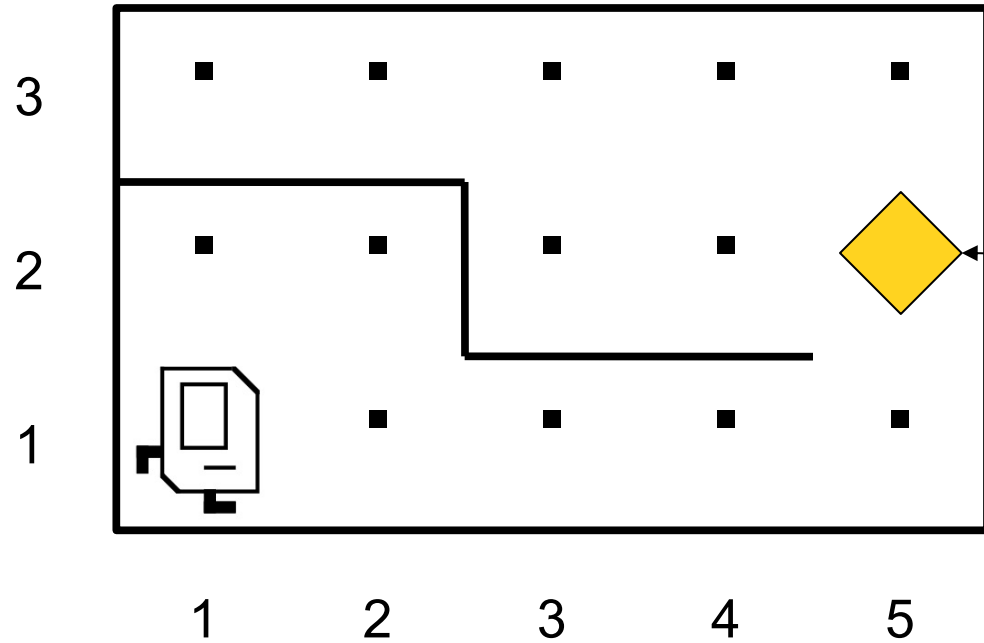
The intersection of a street and an avenue is a corner.

Walls



Karel cannot
move through
walls.

Beepers



Beepers mark locations in Karel's world. Karel can pick them up and put them down.

Karel Knows 4 Commands



`move`

`turn_left`

`put_beeper`

`pick_beeper`

Karel Knows 4 Commands



`move`

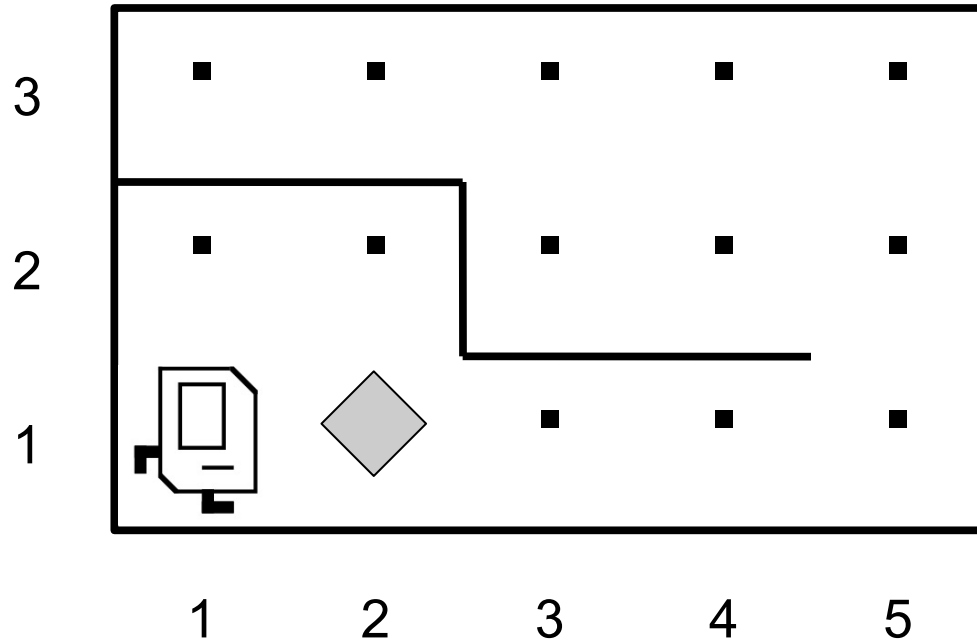
`turn_left`

`put_beeper`

`pick_beeper`

“functions”

Commands: move



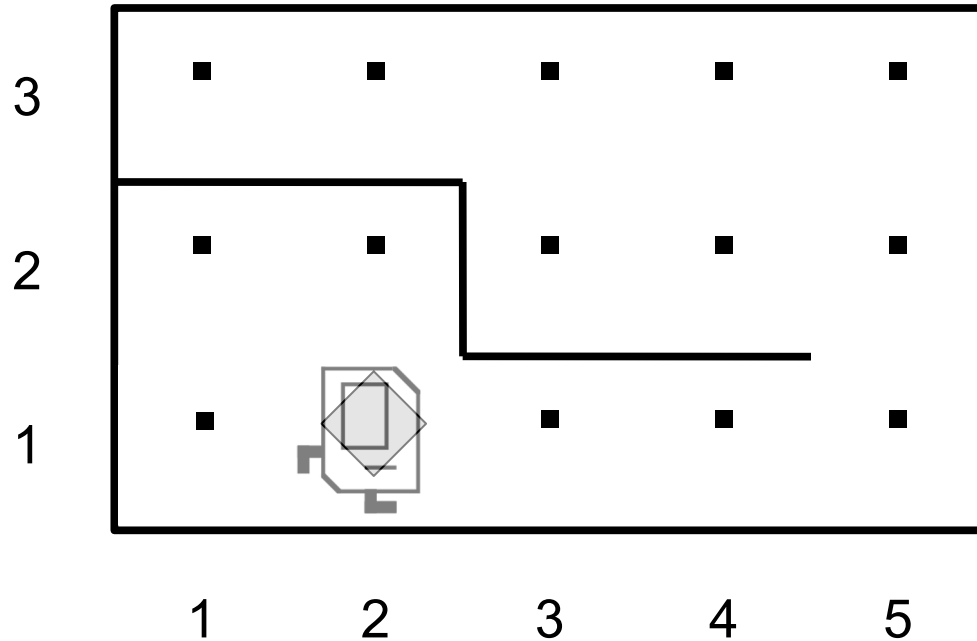
Karel Commands

move

turn_left
pick_beeper
put_beeper

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

Commands: move



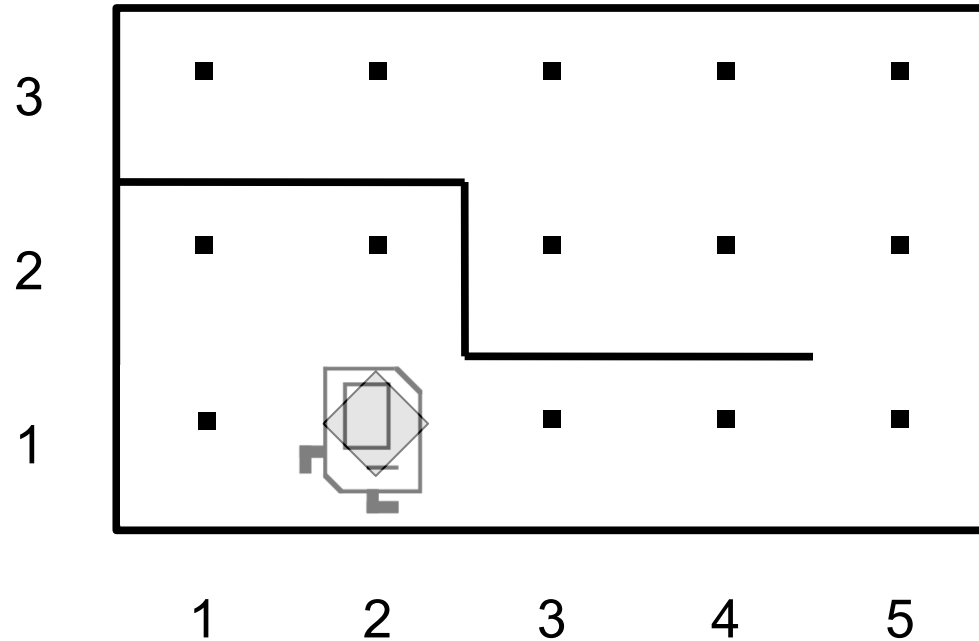
Karel Commands

move

turn_left
pick_beeper
put_beeper

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

Commands: `turn_left`

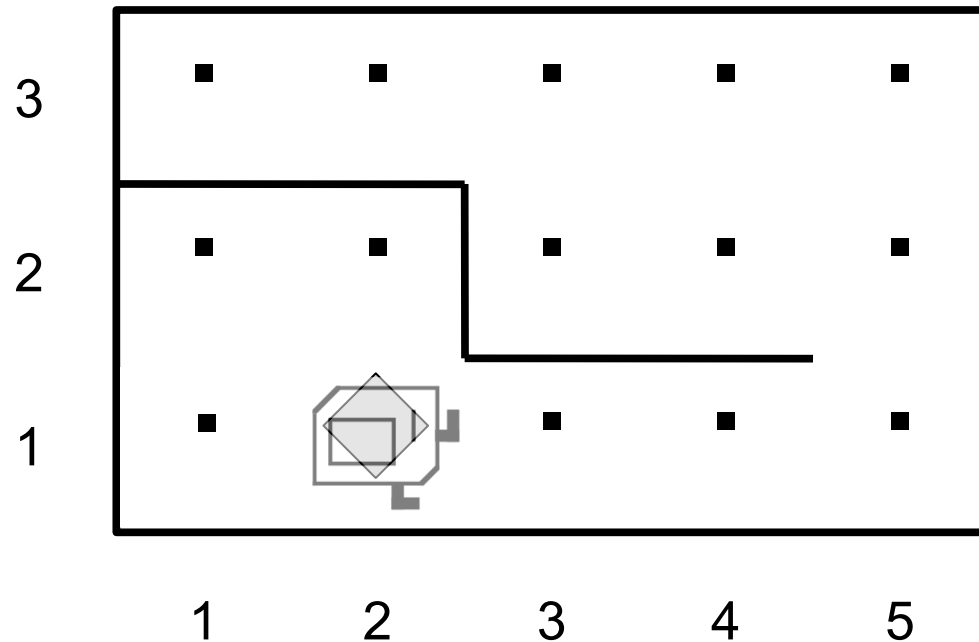


Karel Commands

```
move  
turn_left  
pick_beeper  
put_beeper
```

- `turn_left` makes Karel rotate 90° counter-clockwise.
- There is no `turn_right` command.

Commands: `turn_left`

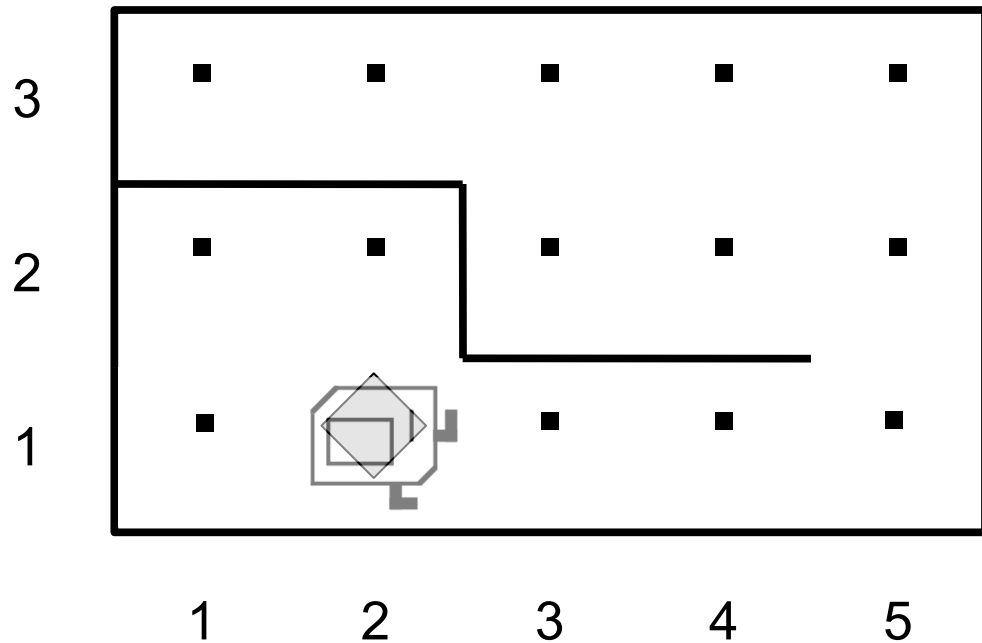


Karel Commands

```
move  
turn_left  
pick_beeper  
put_beeper
```

- `turn_left` makes Karel rotate 90° counter-clockwise.
- There is no `turn_right` command.

Commands: pick_beeper

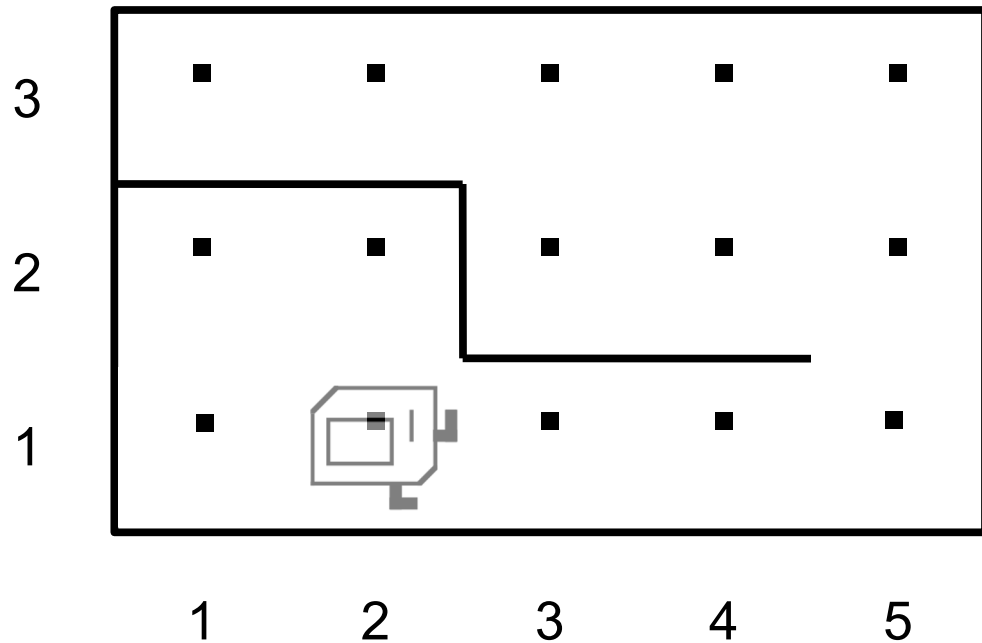


Karel Commands

move
turn_left
pick_beeper
put_beeper

`pick_beeper` makes Karel pick up the beeper at the current corner. Karel can hold multiple beepers at a time in its "beeper bag".

Commands: pick_beeper

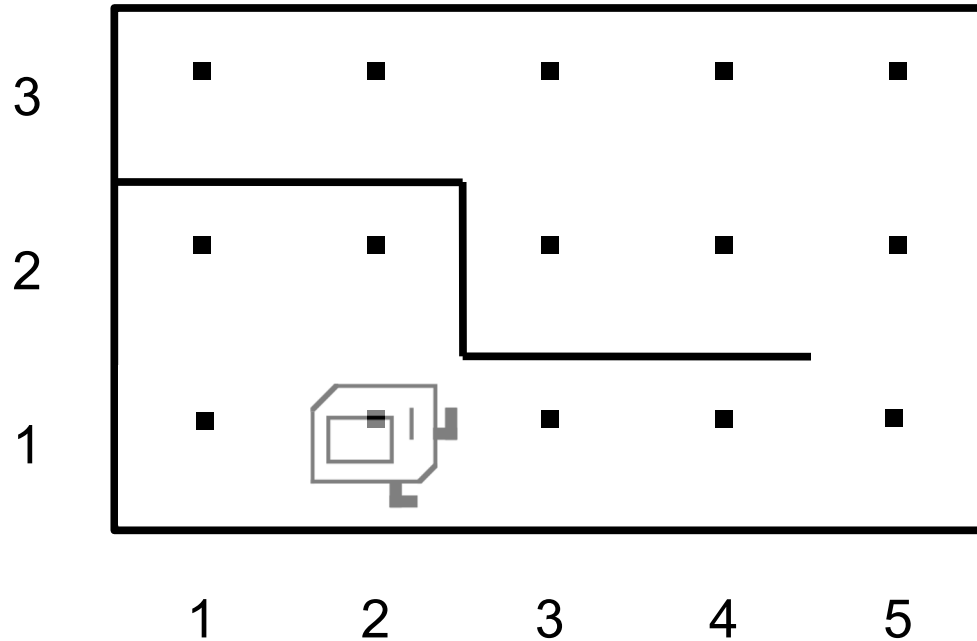


Karel Commands

`move`
`turn_left`
`pick_beeper`
`put_beeper`

`pick_beeper` makes Karel pick up the beeper at the current corner. Karel can hold multiple beepers at a time in its "beeper bag".

Commands: put_beeper



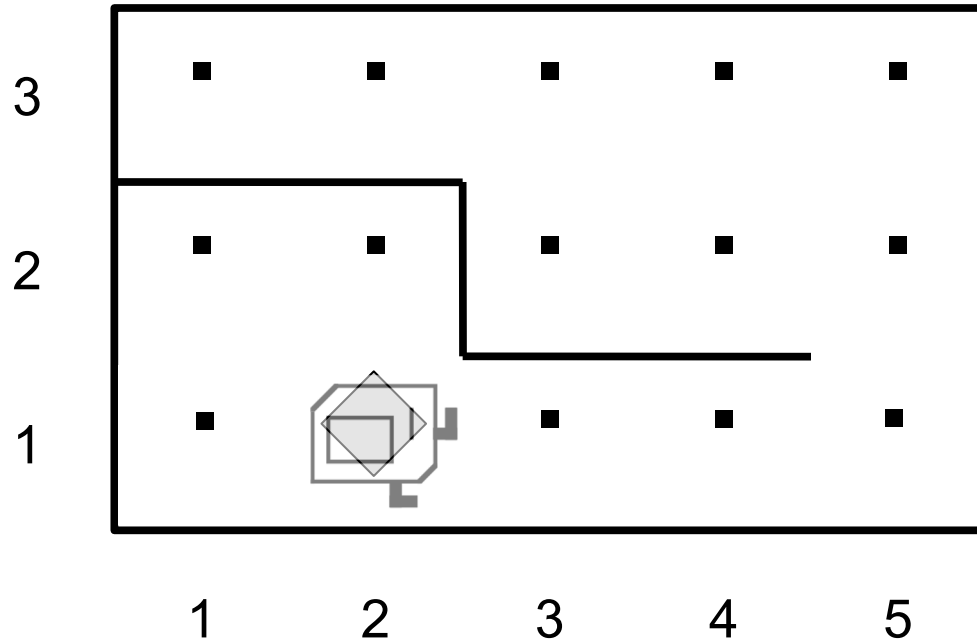
Karel Commands

```
move  
turn_left  
pick_beeper  
put_beeper
```

`put_beeper` makes Karel put a beeper down at its current location.

- `pick_beeper` and `put_beeper` are used to move beepers around.

Commands: put_beeper



Karel Commands

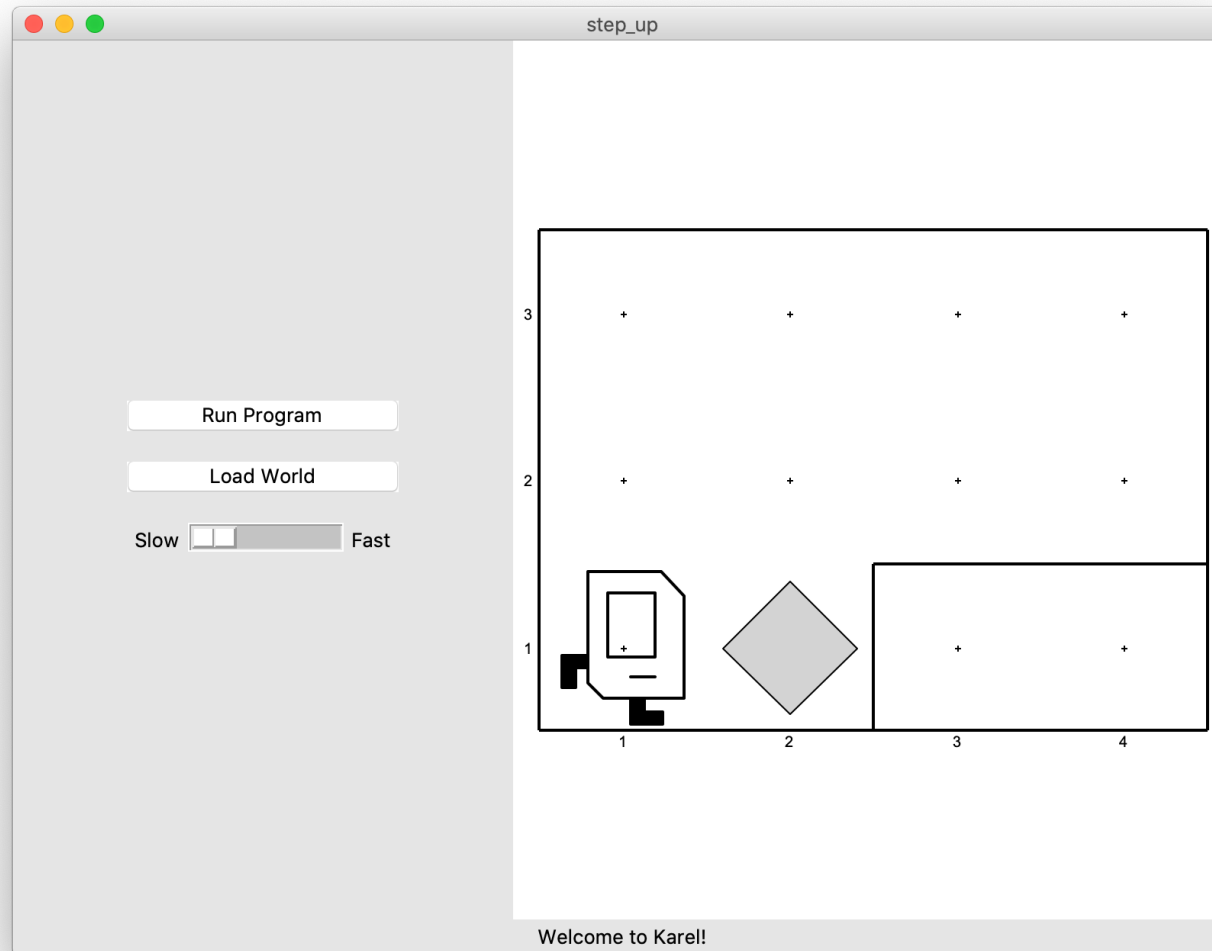
```
move  
turn_left  
pick_beeper  
put_beeper
```

`put_beeper` makes Karel put a beeper down at its current location.

- `pick_beeper` and `put_beeper` are used to move beepers around.

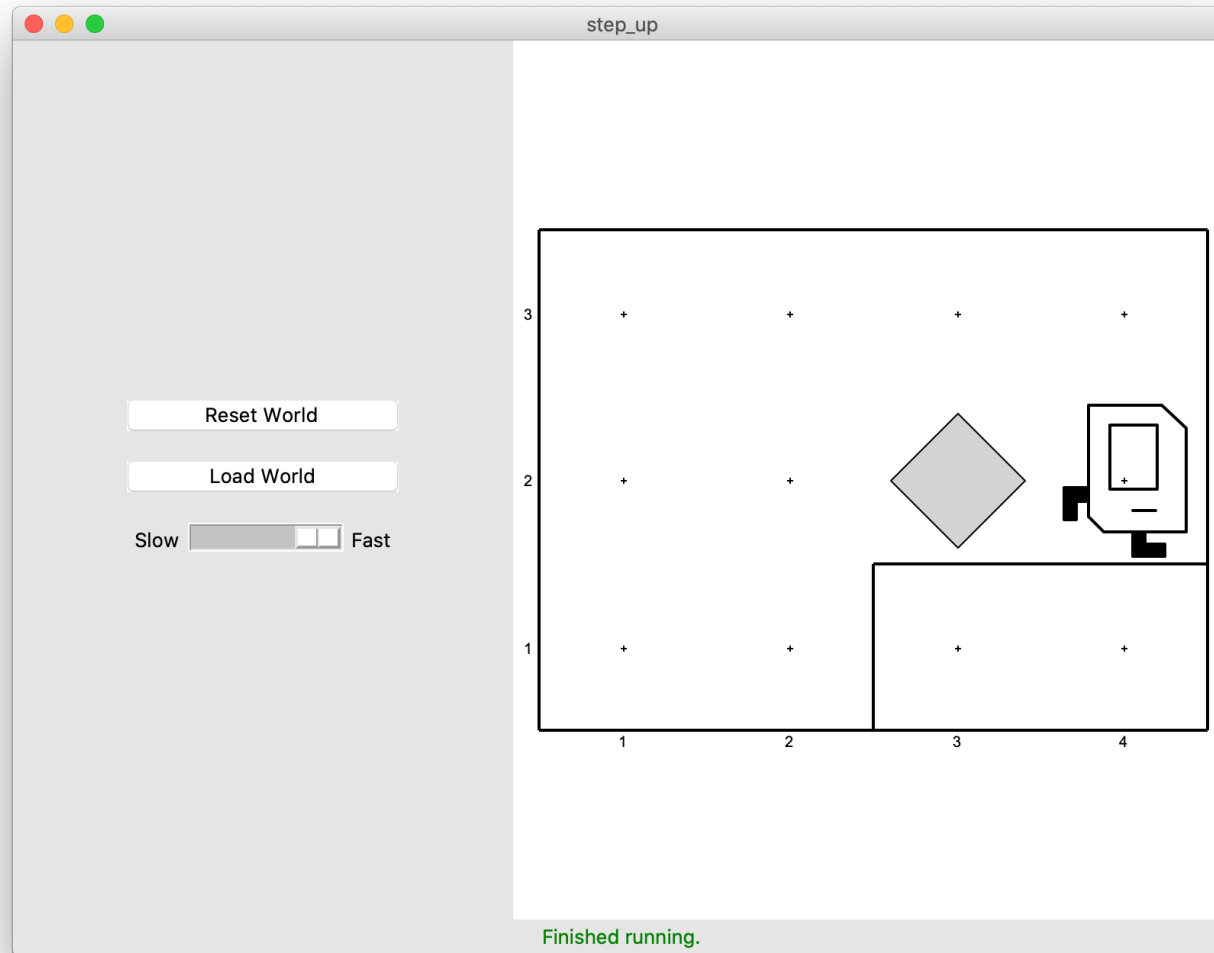
Our First Karel Program

Before:



Our First Karel Program

After:



Demo

Anatomy Of A Program

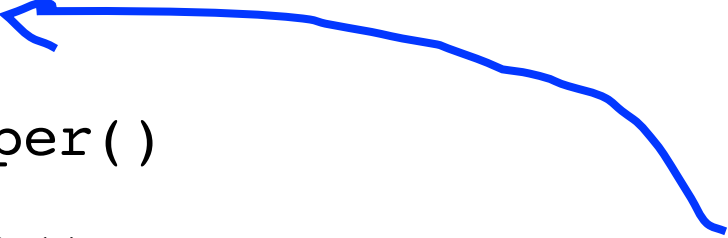
```
def main():  
    move()  
    pick_beeper()  
    move()  
    turn_left()  
    move()  
    turn_left()  
    move()  
    put_beeper()  
    move()  
    ...
```

This piece of the program's
source code is called a
function.

```
if __name__ == "__main__":  
    run_karel_program()
```

Anatomy Of A Program

```
def main():  
    move()  
    pick_beeper()  
    move()  
    turn_left()  
    move()  
    turn_left()  
    move()  
    put_beeper()  
    move()  
    ...
```



This line of code gives the *name* of the function (here, **main**)

```
if __name__ == "__main__":  
    run_karel_program()
```

Anatomy Of A Program

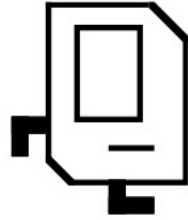
```
def main():  
    move()  
    pick_beeper()  
    move()  
    turn_left()  
    move()  
    turn_left()  
    move()  
    put_beeper()  
    move()  
    ...
```

This is called a *code block*. Python requires that code blocks be indented one level.

```
if __name__ == "__main__":  
    run_karel_program()
```

Lecture Plan

- Welcome to CS Bridge!
- Course information
- Meet Karel the Robot
- **Defining new commands**
- For loops



Defining New Commands

We can make new commands (or **functions**) for Karel. This lets us *decompose* our program into smaller pieces that are easier to understand.

```
def name():  
    statement  
    statement  
    ...
```

For example:

```
def turn_right():  
    turn_left()  
    turn_left()  
    turn_left()
```

Decomposition

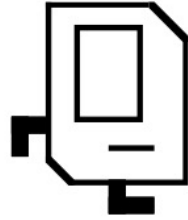
- Breaking down problems into smaller, more approachable sub-problems (e.g. our own Karel commands)
- Each piece should solve **one** problem/task (< ~ 20 lines of code)
 - Descriptively-named
 - Well-commented!
- E.g. getting up in the morning:
 - Wake up
 - Brush teeth
 - Put toothpaste on toothbrush
 - Insert toothbrush into mouth
 - Move toothbrush against teeth
 - ...
 - ...

Top-Down Design

- Start from a large task and break it up into smaller pieces
- **Goal:** make our programs easily readable by humans
 - Commenting
 - Decomposition

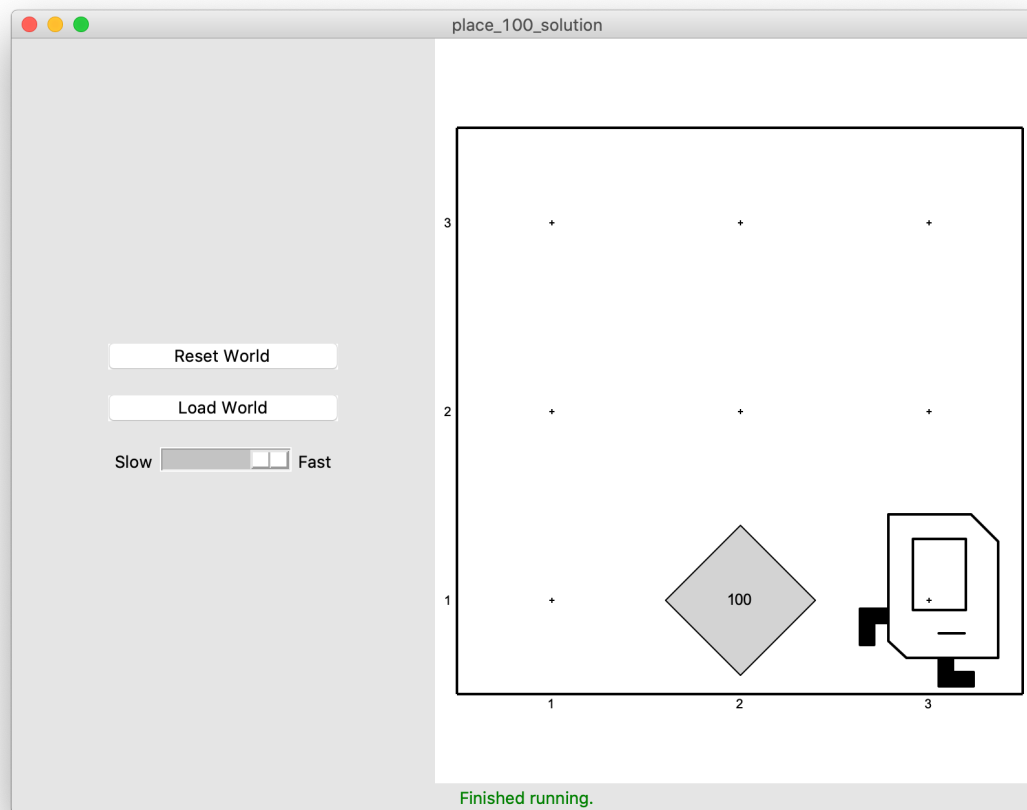
Lecture Plan

- Welcome to CS Bridge!
- Course information
- Meet Karel the Robot
- Defining new commands
- **For loops**



For Loops

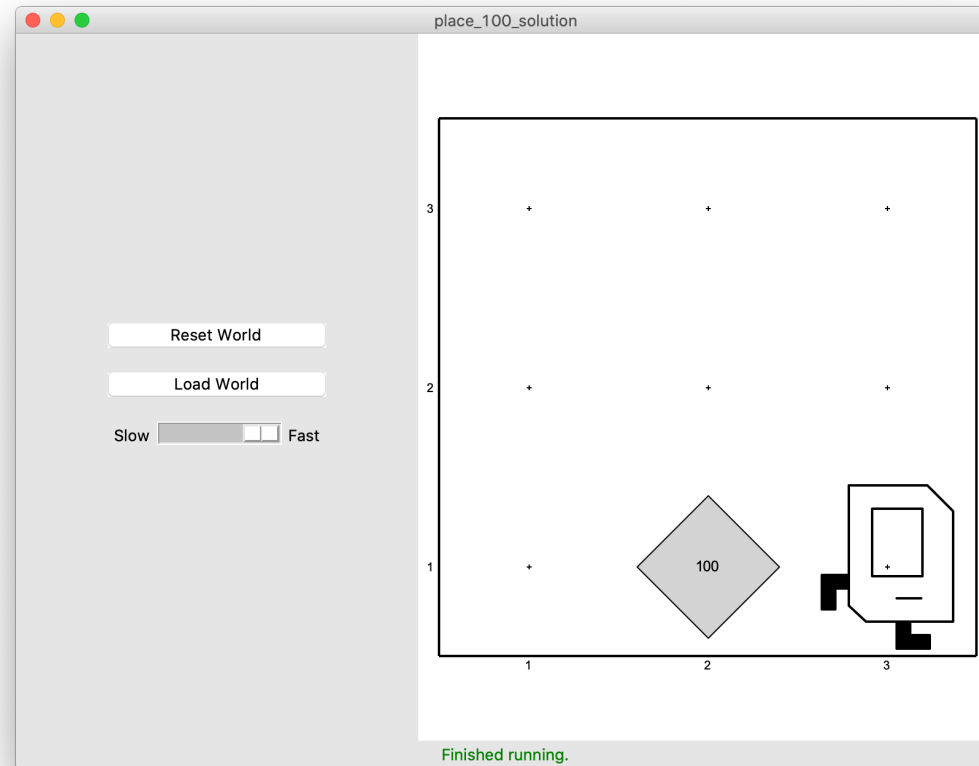
I want to make Karel put 100 beepers down on a corner. How do I do this?



For Loops

Can't just say:

```
move ( )  
put_beeper ( )  
put_beeper ( )  
put_beeper ( )  
...  
move ( )
```



This is too repetitive! Plus, it's difficult to change (e.g. to 25 beepers).

For Loops

Instead, use a **for** loop:

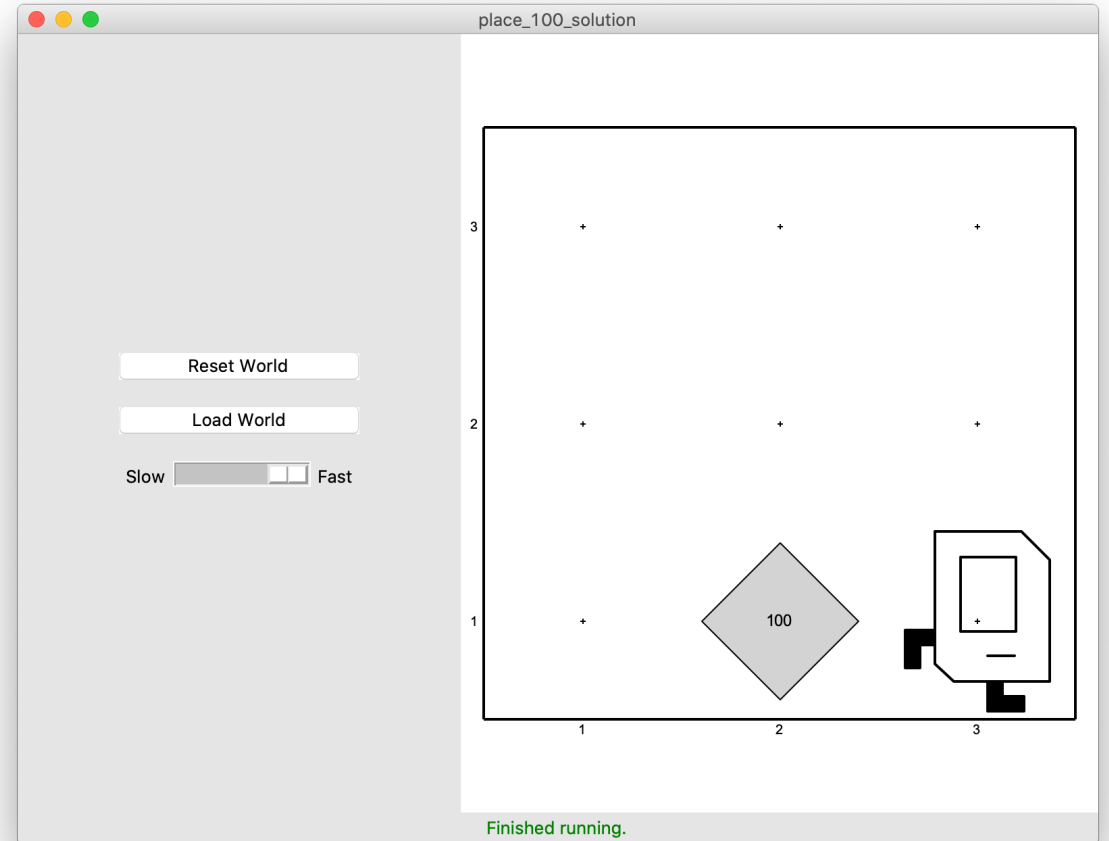
```
for i in range(count):  
    statement  
    statement  
    ...
```

Repeats the statements in the body **count** times.

For Loops


Now we can say:


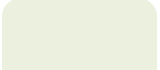
```
move()  
for i in range(100):  
    put_beeper()  
move()
```



This is less repetitive and is easier to change (e.g. to 25 beepers).

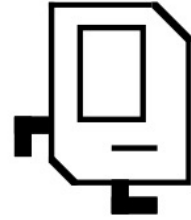
For Loops and Indentation

```
for i in range(count):  
     statements           # note indenting
```

```
def turn_right():  
     for i in range(3):  
         turn_left()           # note indenting
```

Lecture Plan

- Welcome to CS Bridge!
- Course information
- Meet Karel the Robot
- Defining new commands
- For loops



Welcome to CS Bridge!

- We are here to share with you our love for programming!
- A joint effort
- **Our goal:** form a community of people to learn and teach programming together



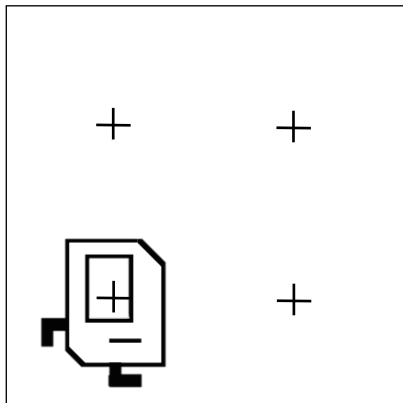
What's Next?

- Time for your section's quickstart time!
- Check your section's Ed group for more information
- We hope you will be able to get set up if you aren't already, and complete at least collect newspaper Karel. Have fun!

Extra: Beeper Square

For Loop Example: Beeper Square

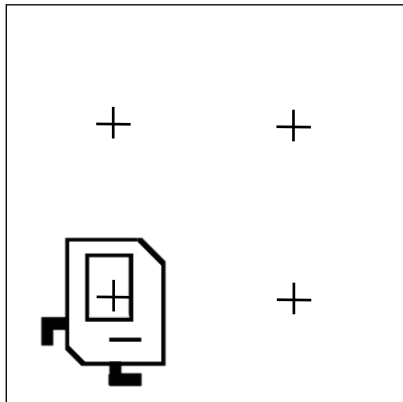
```
def main():  
    for i in range(4):  
        put_beeper()  
        move()  
        turn_left()
```



For Loop Example: Beeper Square

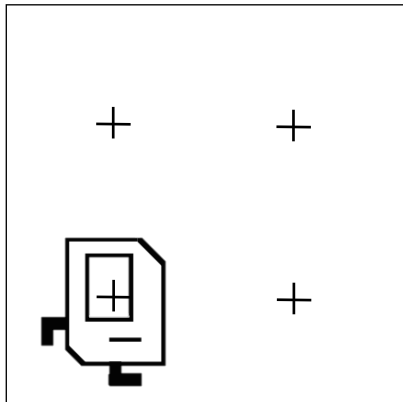
```
def main():
```

```
    for i in range(4):  
        put_beeper()  
        move()  
        turn_left()
```



For Loop Example: Beeper Square

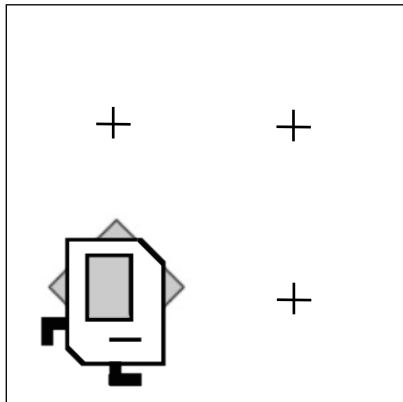
```
def main():  
    for i in range(4):  
        put_beeper()  
        move()  
        turn_left()
```



First time
through the
loop

For Loop Example: Beeper Square

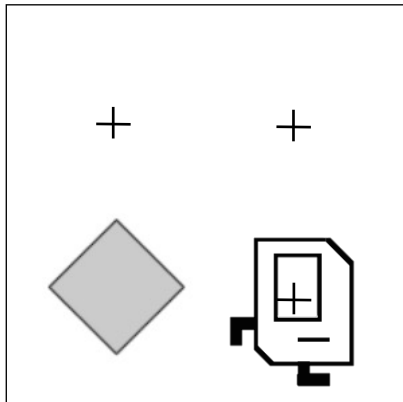
```
def main():  
    for i in range(4):  
        put_beeper()  
        move()  
        turn_left()
```



First time
through the
loop

For Loop Example: Beeper Square

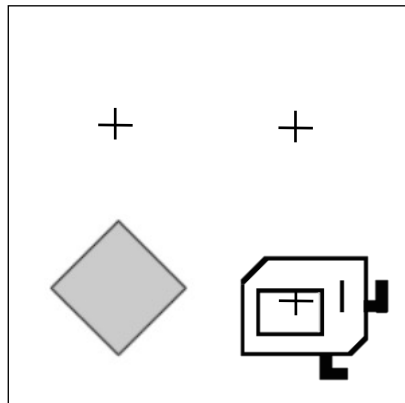
```
def main():  
    for i in range(4):  
        put beeper()  
        move()  
        turn_left()
```



First time
through the
loop

For Loop Example: Beeper Square

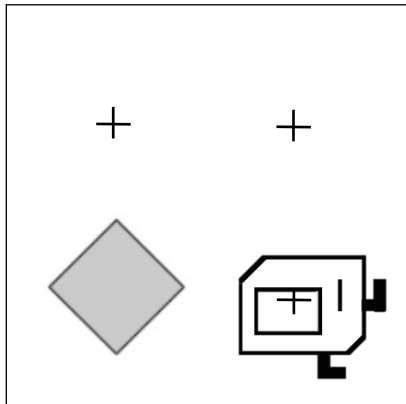
```
def main():  
    for i in range(4):  
        put_beeper()  
        move()  
        turn left()
```



First time
through the
loop

For Loop Example: Beeper Square

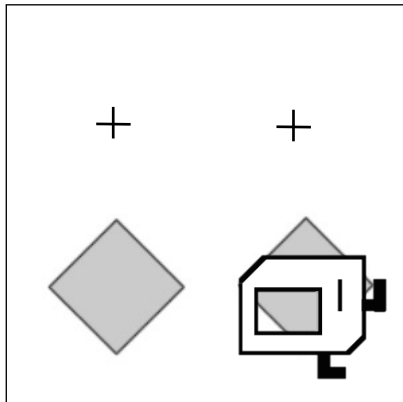
```
def main():  
    for i in range(4):  
        put_beeper()  
        move()  
        turn_left()
```



Second time
through the
loop

For Loop Example: Beeper Square

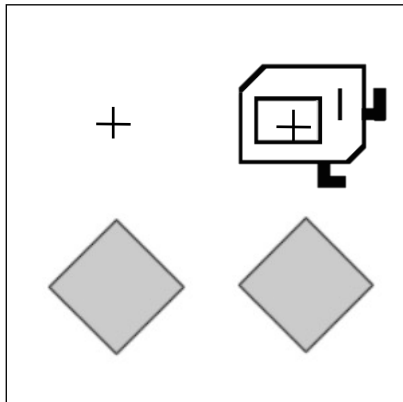
```
def main():  
    for i in range(4):  
        put_beeper()  
        move()  
        turn_left()
```



Second time
through the
loop

For Loop Example: Beeper Square

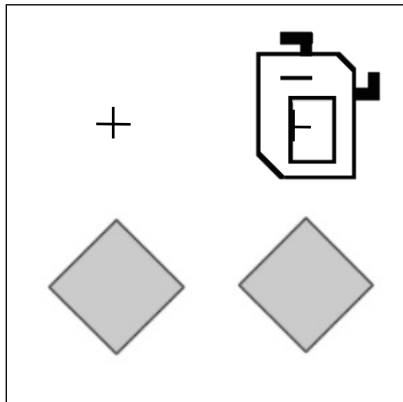
```
def main():  
    for i in range(4):  
        put beeper()  
        move()  
        turn_left()
```



Second time
through the
loop

For Loop Example: Beeper Square

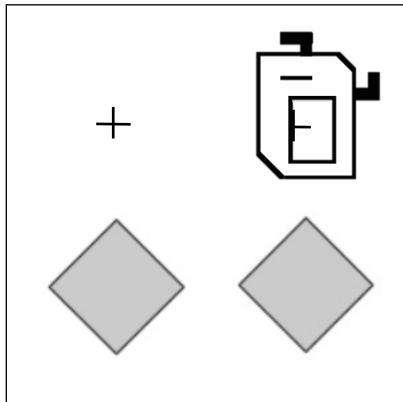
```
def main():  
    for i in range(4):  
        put_beeper()  
        move()  
        turn left()
```



Second time
through the
loop

For Loop Example: Beeper Square

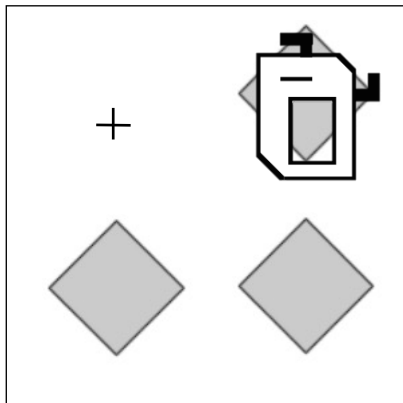
```
def main():  
    for i in range(4):  
        put_beeper()  
        move()  
        turn_left()
```



Third time
through the
loop

For Loop Example: Beeper Square

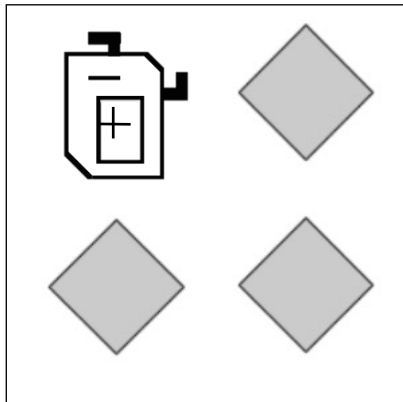
```
def main():  
    for i in range(4):  
        put_beeper()  
        move()  
        turn_left()
```



Third time
through the
loop

For Loop Example: Beeper Square

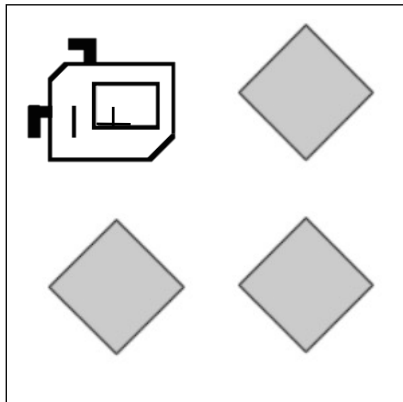
```
def main():  
    for i in range(4):  
        put beeper()  
        move()  
        turn_left()
```



Third time
through the
loop

For Loop Example: Beeper Square

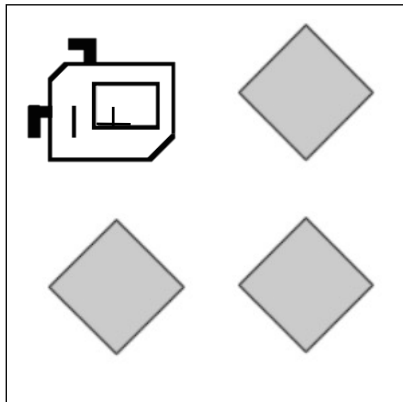
```
def main():  
    for i in range(4):  
        put_beeper()  
        move()  
        turn left()
```



Third time
through the
loop

For Loop Example: Beeper Square

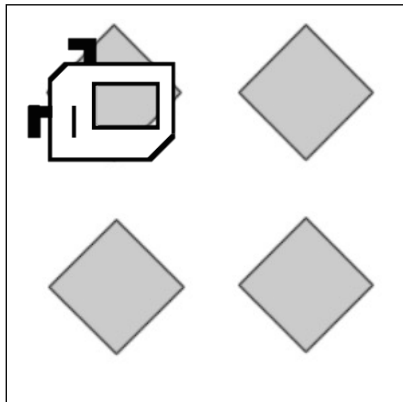
```
def main():  
    for i in range(4):  
        put_beeper()  
        move()  
        turn_left()
```



Fourth time
through the
loop

For Loop Example: Beeper Square

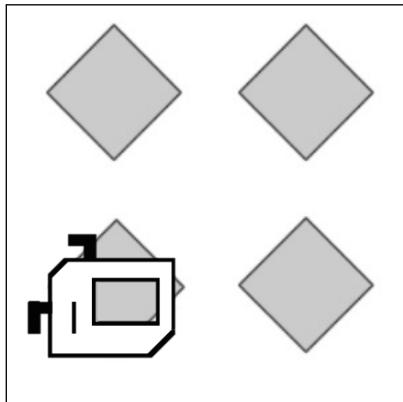
```
def main():  
    for i in range(4):  
        put_beeper()  
        move()  
        turn_left()
```



Fourth time
through the
loop

For Loop Example: Beeper Square

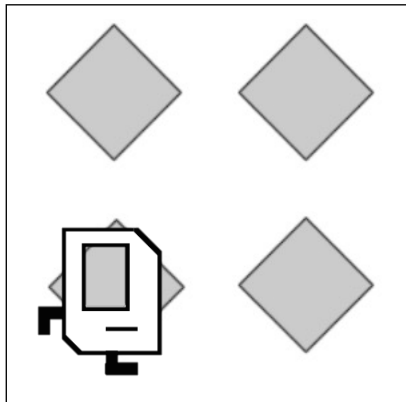
```
def main():  
    for i in range(4):  
        put beeper()  
        move()  
        turn_left()
```



Fourth time
through the
loop

For Loop Example: Beeper Square

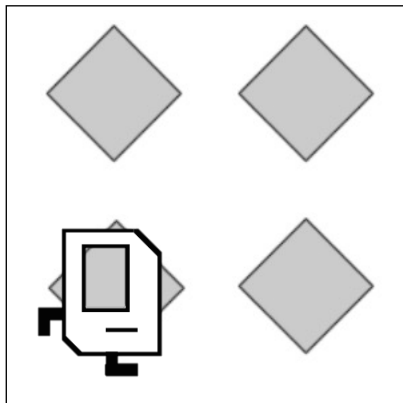
```
def main():  
    for i in range(4):  
        put_beeper()  
        move()  
        turn_left()
```



Fourth time
through the
loop

For Loop Example: Beeper Square

```
def main():  
    for i in range(4):  
        put_beeper()  
        move()  
        turn_left()
```



Done!