

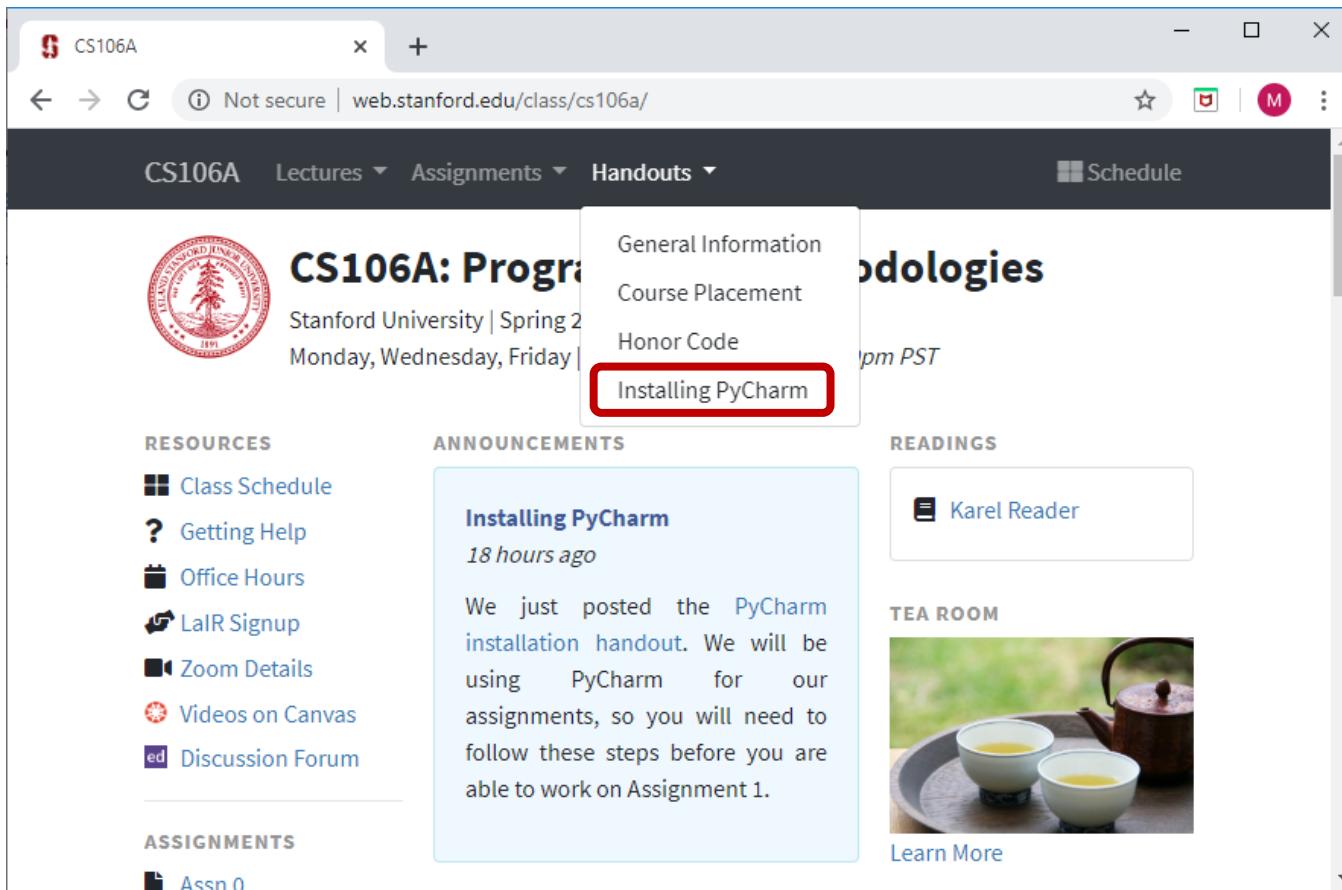
# Control Flow

Chris Gregg

CS106A, Stanford University

Modified from slides courtesy of  
Chris Piech and Mehran Sahami

# Install PyCharm



CS106A

Not secure | web.stanford.edu/class/cs106a/

CS106A Lectures Assignments Handouts Schedule

General Information Course Placement Honor Code **Installing PyCharm**

RESOURCES

- Class Schedule
- Getting Help
- Office Hours
- LaIR Signup
- Zoom Details
- Videos on Canvas
- Discussion Forum

ANNOUNCEMENTS

**Installing PyCharm**  
18 hours ago

We just posted the [PyCharm installation handout](#). We will be using PyCharm for our assignments, so you will need to follow these steps before you are able to work on Assignment 1.

READINGS

Karel Reader

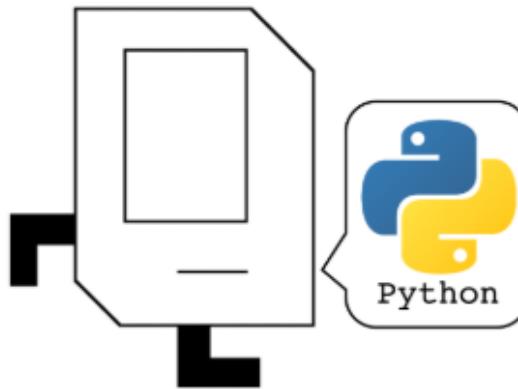
TEA ROOM

Learn More

Please follow instructions *closely*.  
Email Brahms if you have problems.



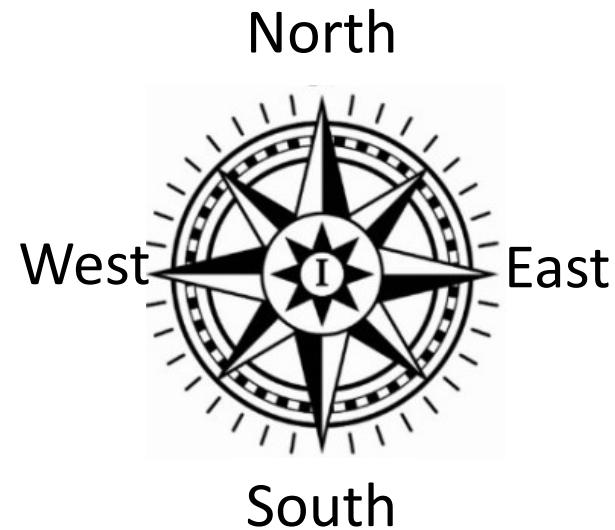
# Using Karel and Assignment 1



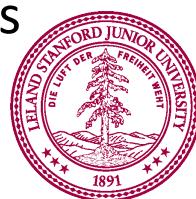
- Reading: Should read the “Karel Reader” on class website
- Handout #3: “Honor Code”
- Handout #4: “Using Karel with PyCharm”
  - Tells you how to get started with writing Karel programs
- Handout #5: “Assignment 1”
  - Set of Karel programs for you to write
  - Due 10:30am (PDT) on Tuesday, June 30th
- Only use features of Karel in the course reader
  - No other features of Python may be used in Karel programs!



# Recall, Karel's World



- Grid, where "corner" is intersection of each street/avenue
- Karel is currently on corner (1, 1)
- If Karel moved forward, Karel would be on corner (2, 1)
- Karel's beeper bag can have 0, 1, or more (up to infinite) beepers



# First Lesson in Programming Style

```
from karel.stanfordkarel import *
```

```
"""
```

File: StepUpKarel.py

```
-----
```

Karel program, where Karel picks up a beeper,  
jumps up on a step and drops the beeper off.

```
"""
```

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

# Karel turns to the right

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

}

Multi-line  
**comment**

**SOFTWARE ENGINEERING PRINCIPLE:**  
Aim to make programs readable by *humans*

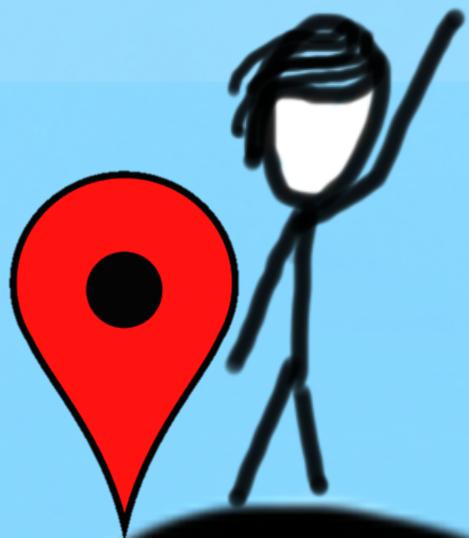
One line  
**comment**

Descriptive  
**names**  
(snake\_case)

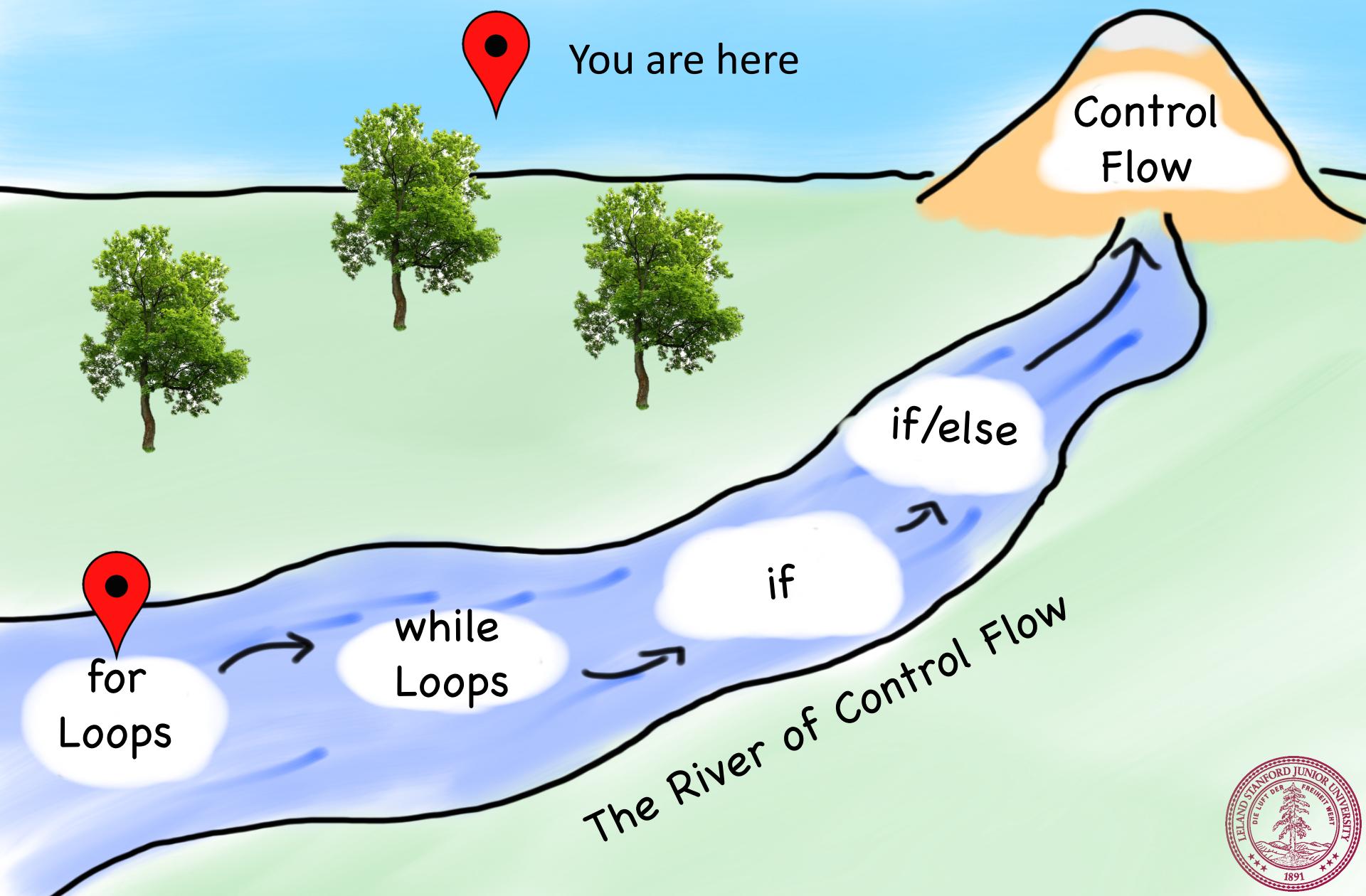


# Today's Goal

1. Code using loops and conditions
2. Trace programs that use loops and conditions



# Today's Route



# for loop

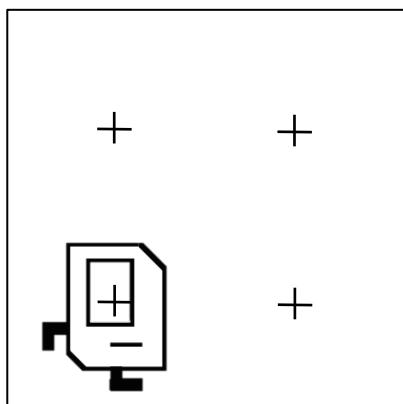
```
for i in range(count) :  
    statements                      # note indenting
```

---

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

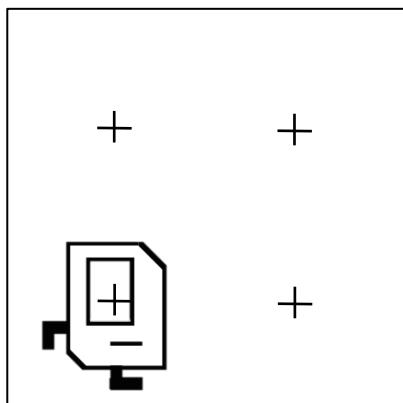
# Place Beeper Square

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



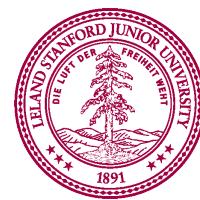
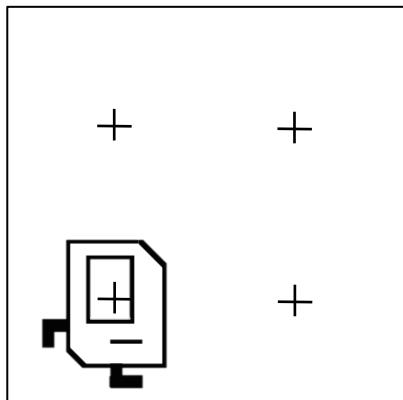
# Place Beeper Square

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



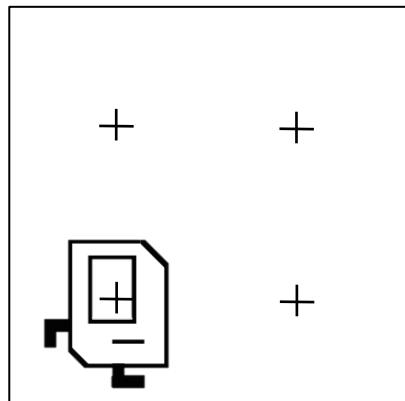
# Place Beeper Square

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



# Place Beeper Square

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

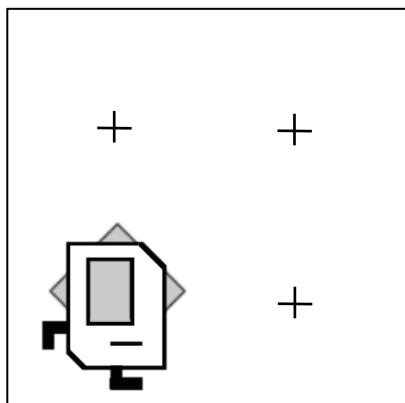


First time  
through the  
loop



# Place Beeper Square

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

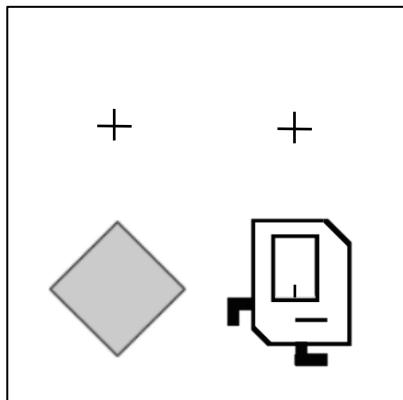


First time  
through the  
loop

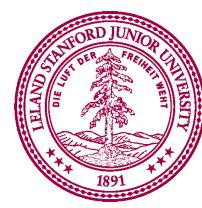


# Place Beeper Square

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

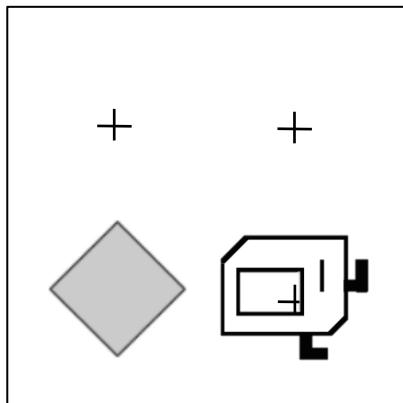


First time  
through the  
loop

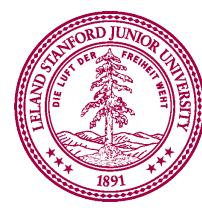


# Place Beeper Square

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

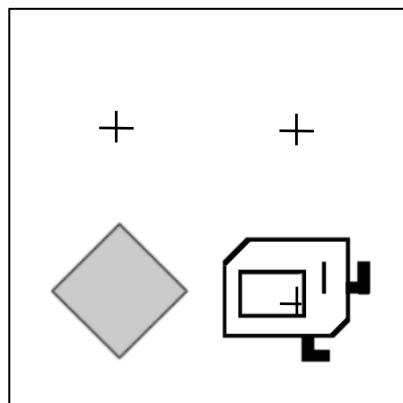


First time  
through the  
loop



# Place Beeper Square

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

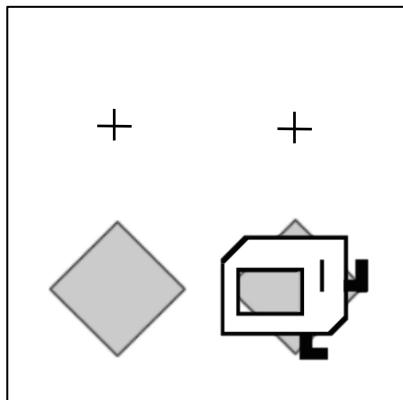


Second time  
through the  
loop



# Place Beeper Square

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

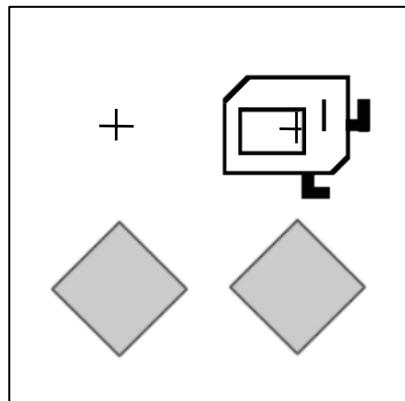


Second time  
through the  
loop



# Place Beeper Square

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

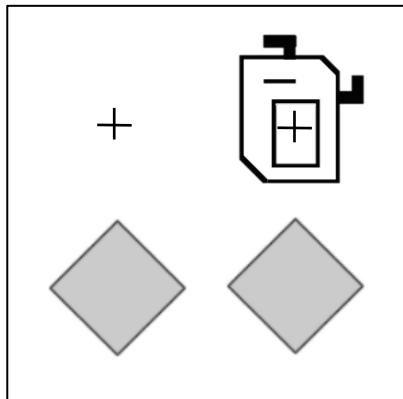


Second time  
through the  
loop



# Place Beeper Square

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

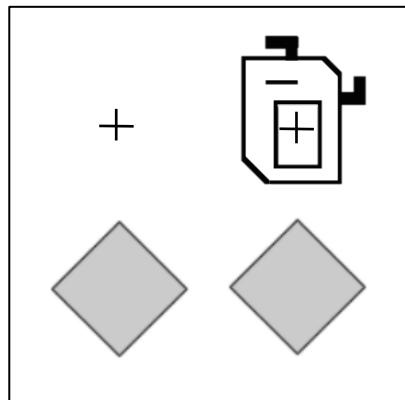


Second time  
through the  
loop



# Place Beeper Square

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

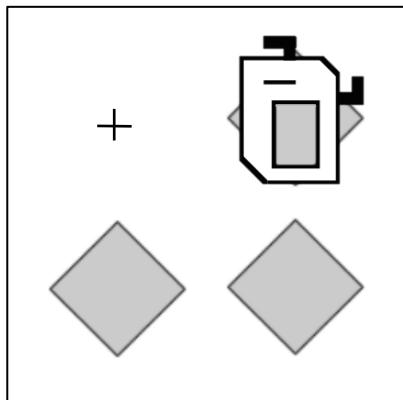


Third time  
through the  
loop



# Place Beeper Square

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

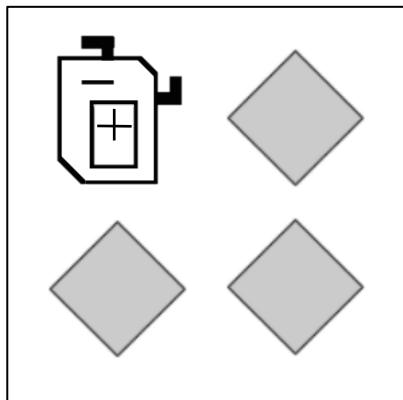


Third time  
through the  
loop

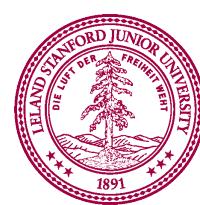


# Place Beeper Square

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

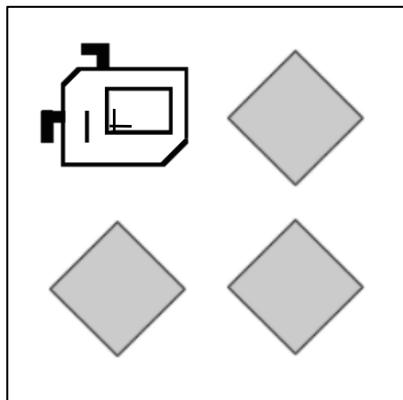


Third time  
through the  
loop



# Place Beeper Square

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

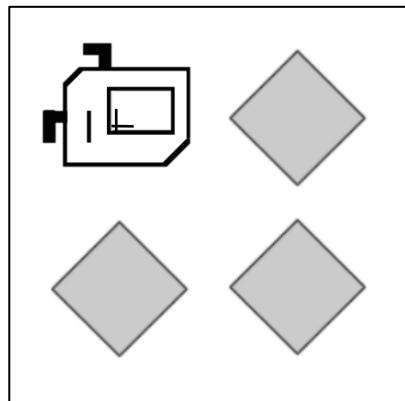


Third time  
through the  
loop



# Place Beeper Square

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

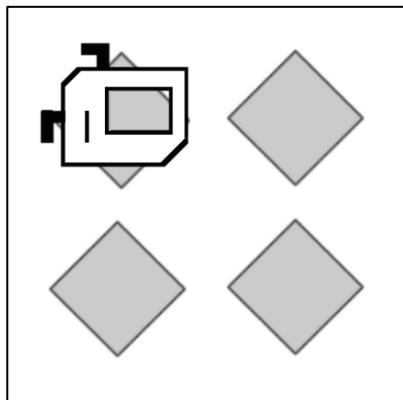


Fourth time  
through the  
loop



# Place Beeper Square

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

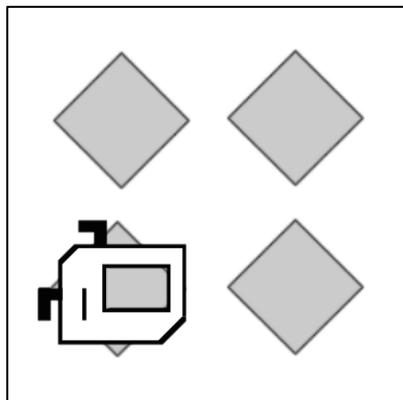


Fourth time  
through the  
loop



# Place Beeper Square

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

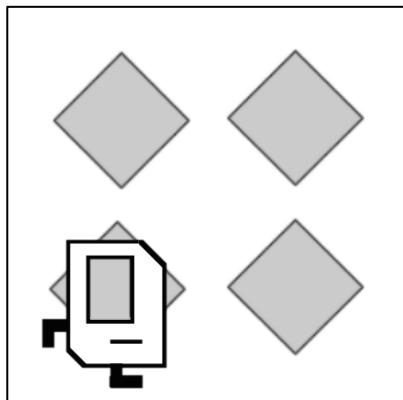


Fourth time  
through the  
loop



# Place Beeper Square

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



Fourth time  
through the  
loop



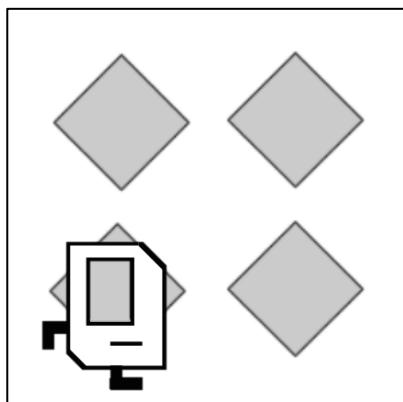
# Place Beeper Square

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

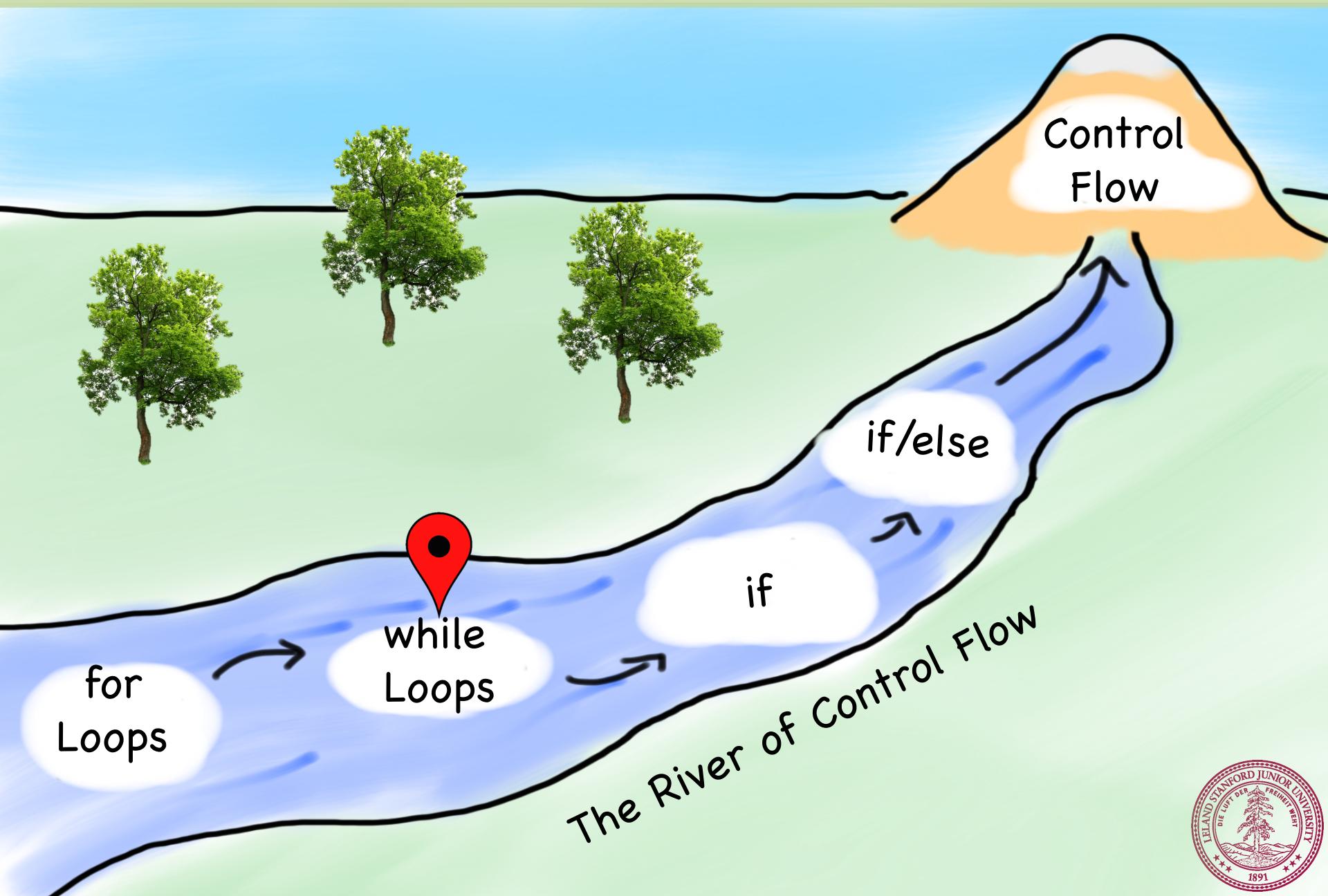


You need the **postcondition** of a loop to match the **precondition**

Done!



# Today's Route



# while loop

```
while condition:
    statements                                # note indenting
```

---

```
def move_to_wall():
    while front_is_clear():
        move()                                # note indenting
```

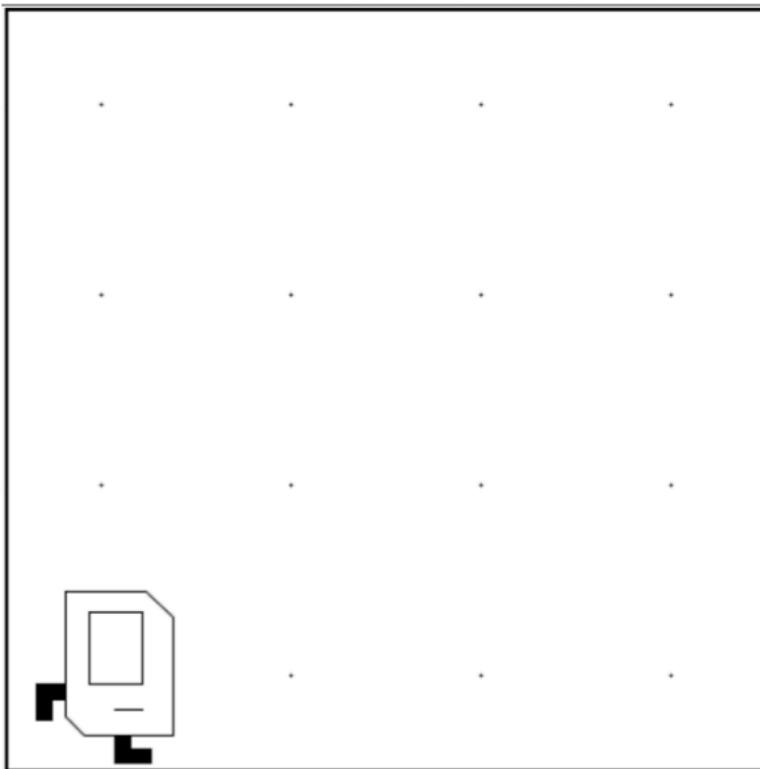
# Conditions Karel Can Check For

<b><i>Test</i></b>	<b><i>Opposite</i></b>	<b><i>What it checks</i></b>
<code>front_is_clear()</code>	<code>front_is_blocked()</code>	Is there a wall in front of Karel?
<code>left_is_clear()</code>	<code>left_is_blocked()</code>	Is there a wall to Karel's left?
<code>right_is_clear()</code>	<code>right_is_blocked()</code>	Is there a wall to Karel's right?
<code>beepers_present()</code>	<code>no_beeper_present()</code>	Are there beepers on this corner?
<code>beepers_in_bag()</code>	<code>no_beeper_in_bag()</code>	Any there beepers in Karel's bag?
<code>facing_north()</code>	<code>not_facing_north()</code>	Is Karel facing north?
<code>facing_east()</code>	<code>not_facing_east()</code>	Is Karel facing east?
<code>facing_south()</code>	<code>not_facing_south()</code>	Is Karel facing south?
<code>facing_west()</code>	<code>not_facing_west()</code>	Is Karel facing west?

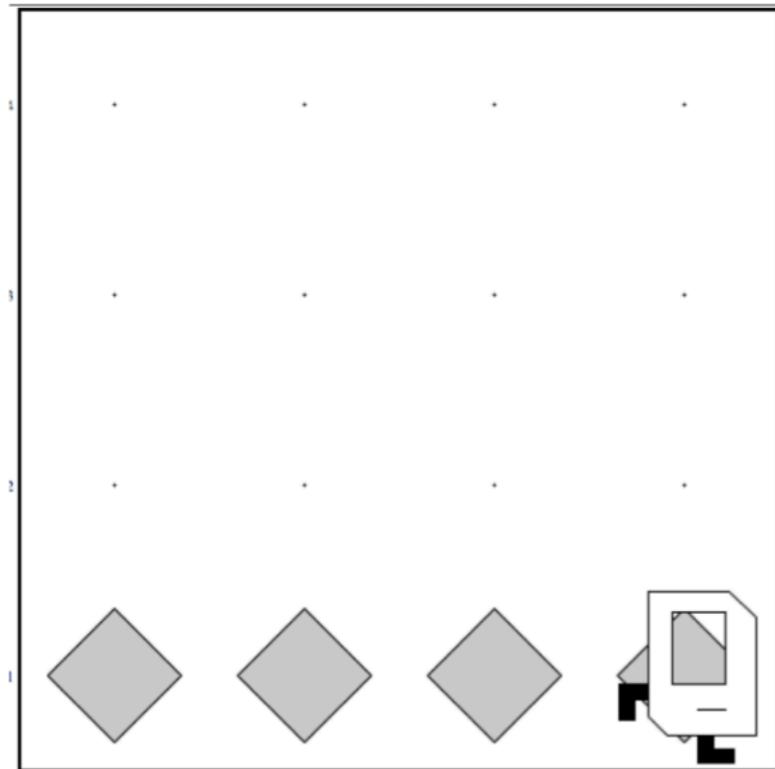
This is in Chapter 10 of the Karel course reader

# Task: Place Beeper Line

Before



After



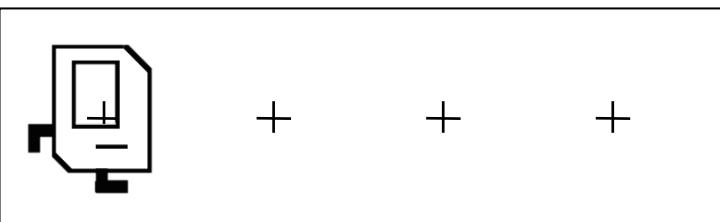
# Place Beeper Line

```
def main():
    while front_is_clear():
        put_beeper()
        move()
```



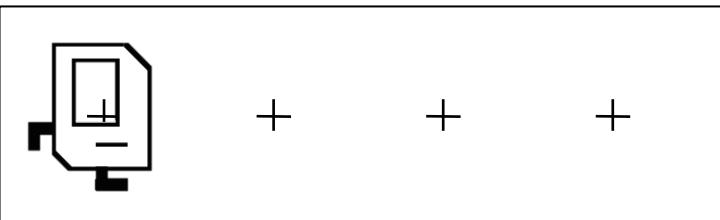
# Place Beeper Line

```
def main():
    while front_is_clear():
        put_beeper()
        move()
```



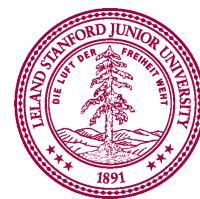
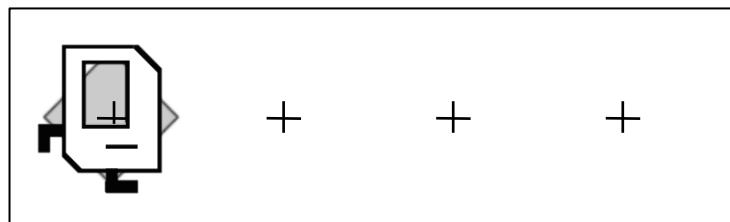
# Place Beeper Line

```
def main():
    while front_is_clear():
        put_beeper()
        move()
```



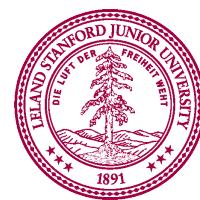
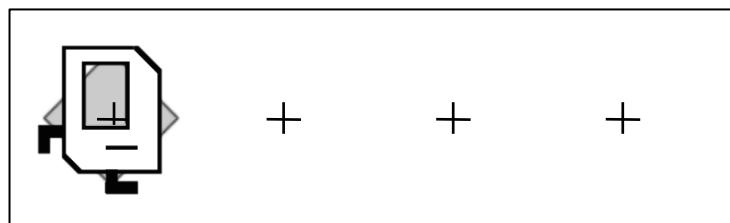
# Place Beeper Line

```
def main():
    while front_is_clear():
        put_beeper()
        move()
```



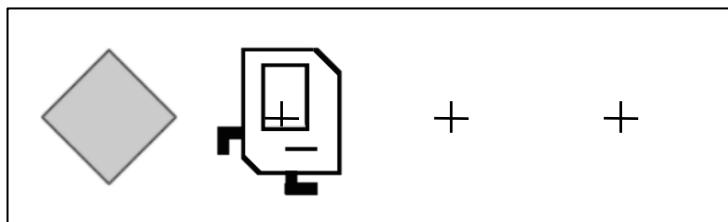
# Place Beeper Line

```
def main():
    while front_is_clear():
        put_beeper()
        move()
```



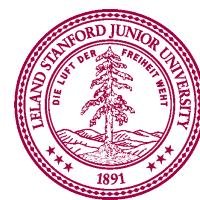
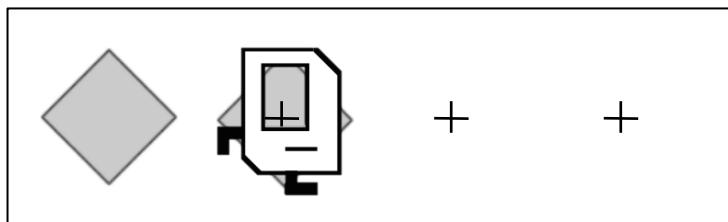
# Place Beeper Line

```
def main():
    while front_is_clear():
        put_beeper()
        move()
```



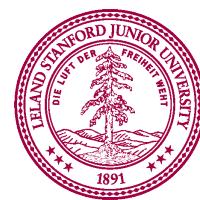
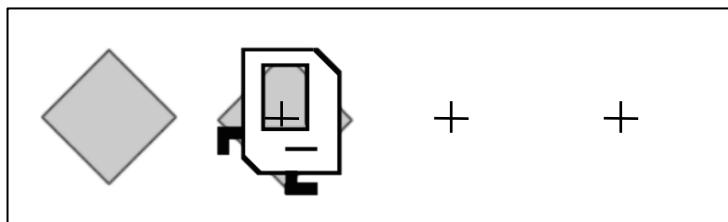
# Place Beeper Line

```
def main():
    while front_is_clear():
        put_beeper()
        move()
```



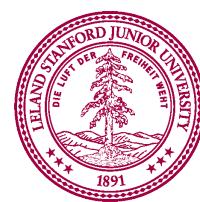
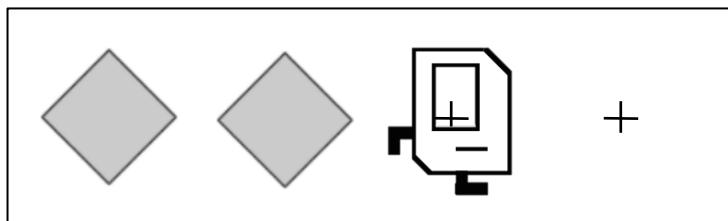
# Place Beeper Line

```
def main():
    while front_is_clear():
        put_beeper()
        move()
```



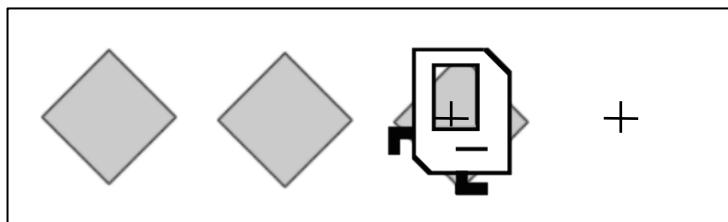
# Place Beeper Line

```
def main():
    while front_is_clear():
        put_beeper()
        move()
```



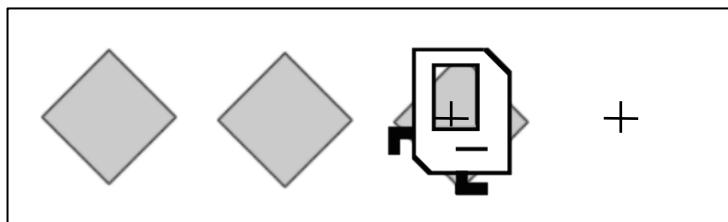
# Place Beeper Line

```
def main():
    while front_is_clear():
        put_beeper()
        move()
```



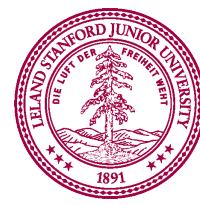
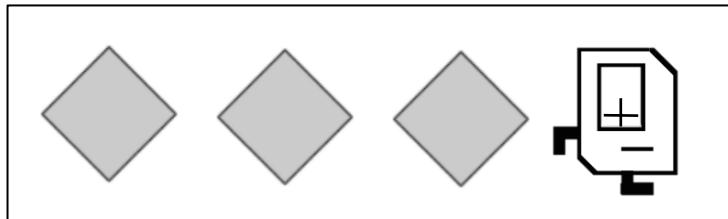
# Place Beeper Line

```
def main():
    while front_is_clear():
        put_beeper()
        move()
```



# Place Beeper Line

```
def main():
    while front_is_clear():
        put_beeper()
        move()
```



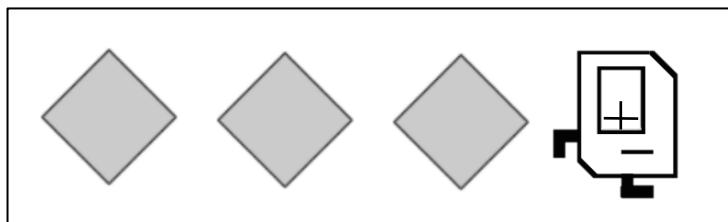
# Place Beeper Line

```
def main():
    while front_is_clear():
        put_beeper()
        move()
```



BUGGY!

Done!



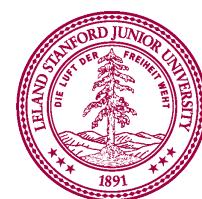
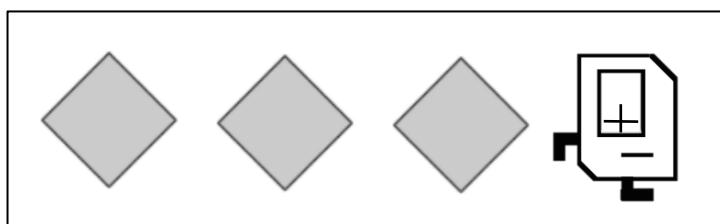
# Place Beeper Line

```
def main():
    while front_is_clear():
        put_beeper()
        move()
    put_beeper() # add final put_beeper
```



Not in **while** loop

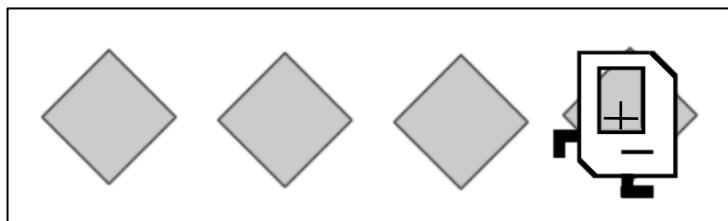
Fixed!



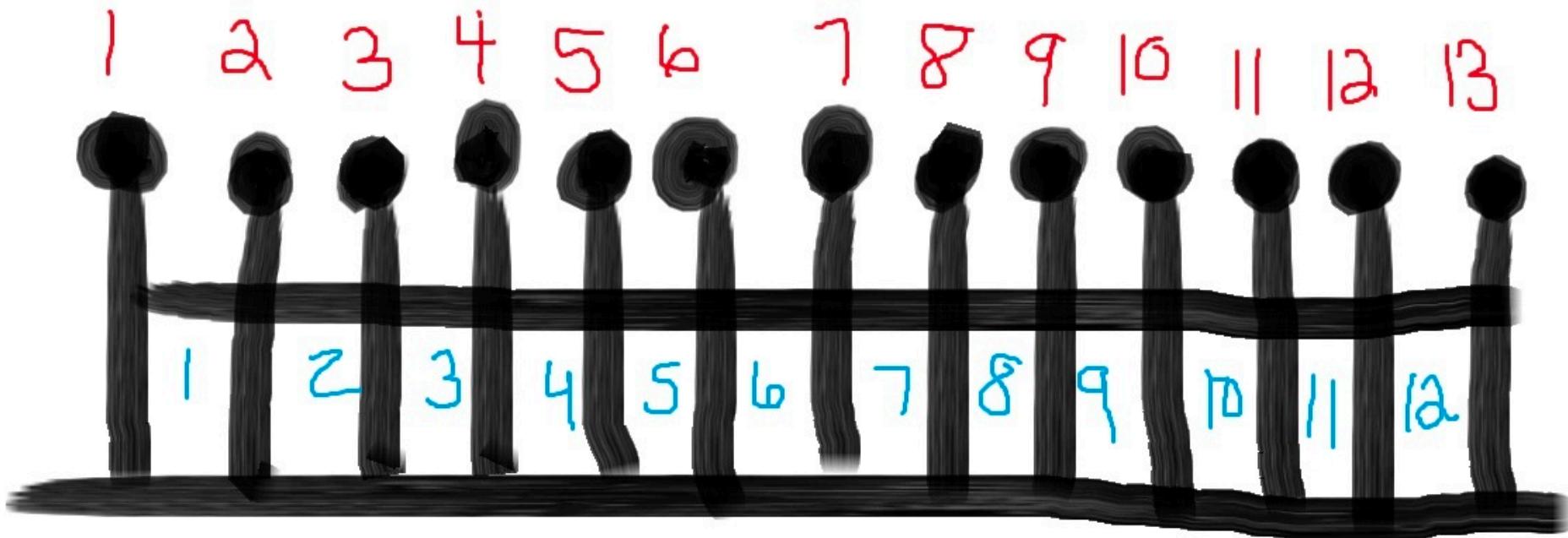
# Place Beeper Line

```
def main():
    while front_is_clear():
        put_beeper()
        move()
    put_beeper() # add final put_beeper
```

Fixed!



# Fence Post Problem



Also sometimes called an “Off By One Error”



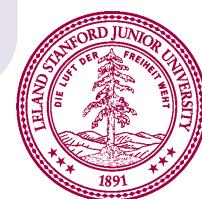
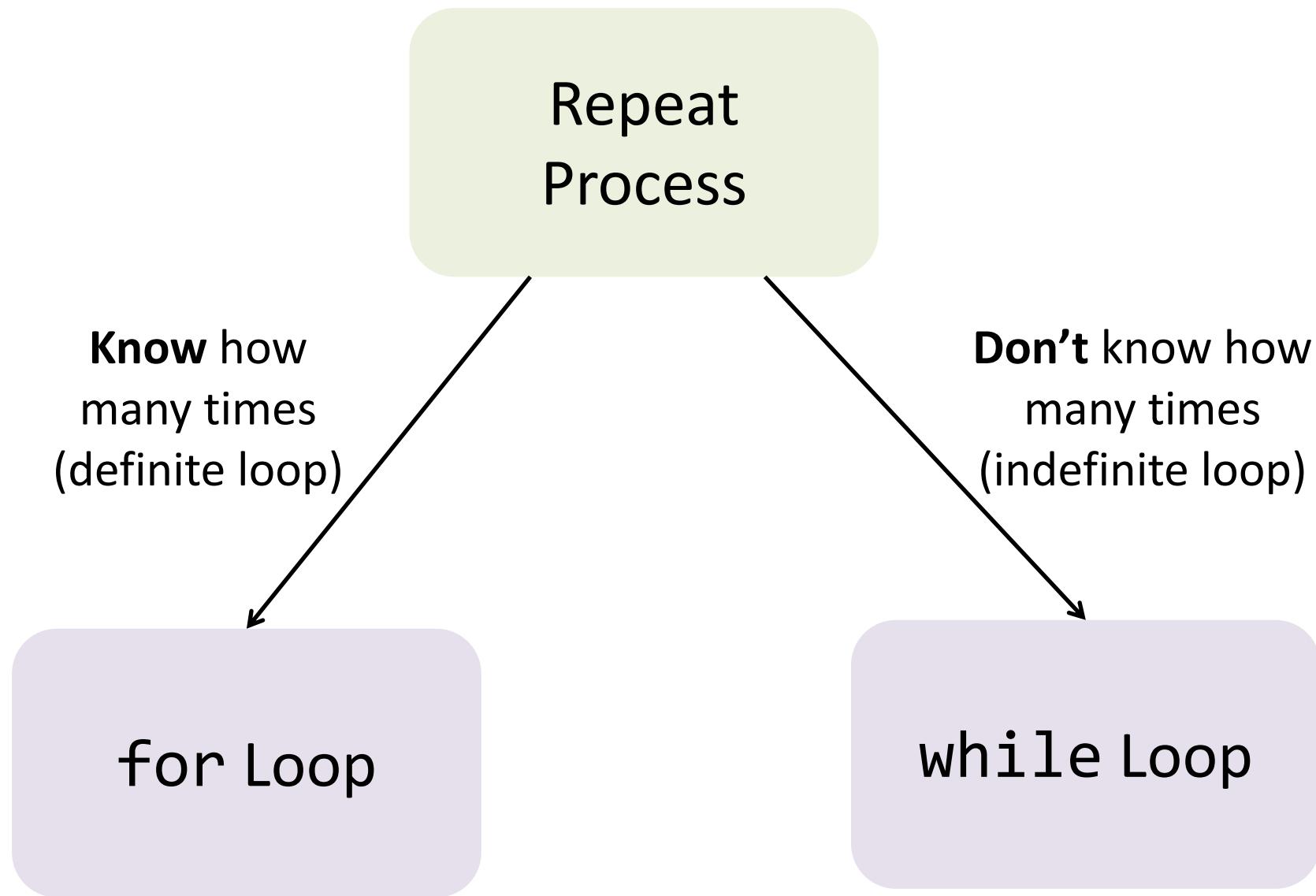


A program executes one line at a time.

The **while** loop checks its condition only at the start of the code block and before repeating.



# Which Loop



# Actual Bug from Marc II

1100      Started Cosine Tap<sup>es</sup> (Sine c  
in relay      Relays changed  
1525      Started Multi Adder Test.

1545



Relay #70  
(moth) in re

First actual case of bug

~~1630~~ 1630 Antilog start.

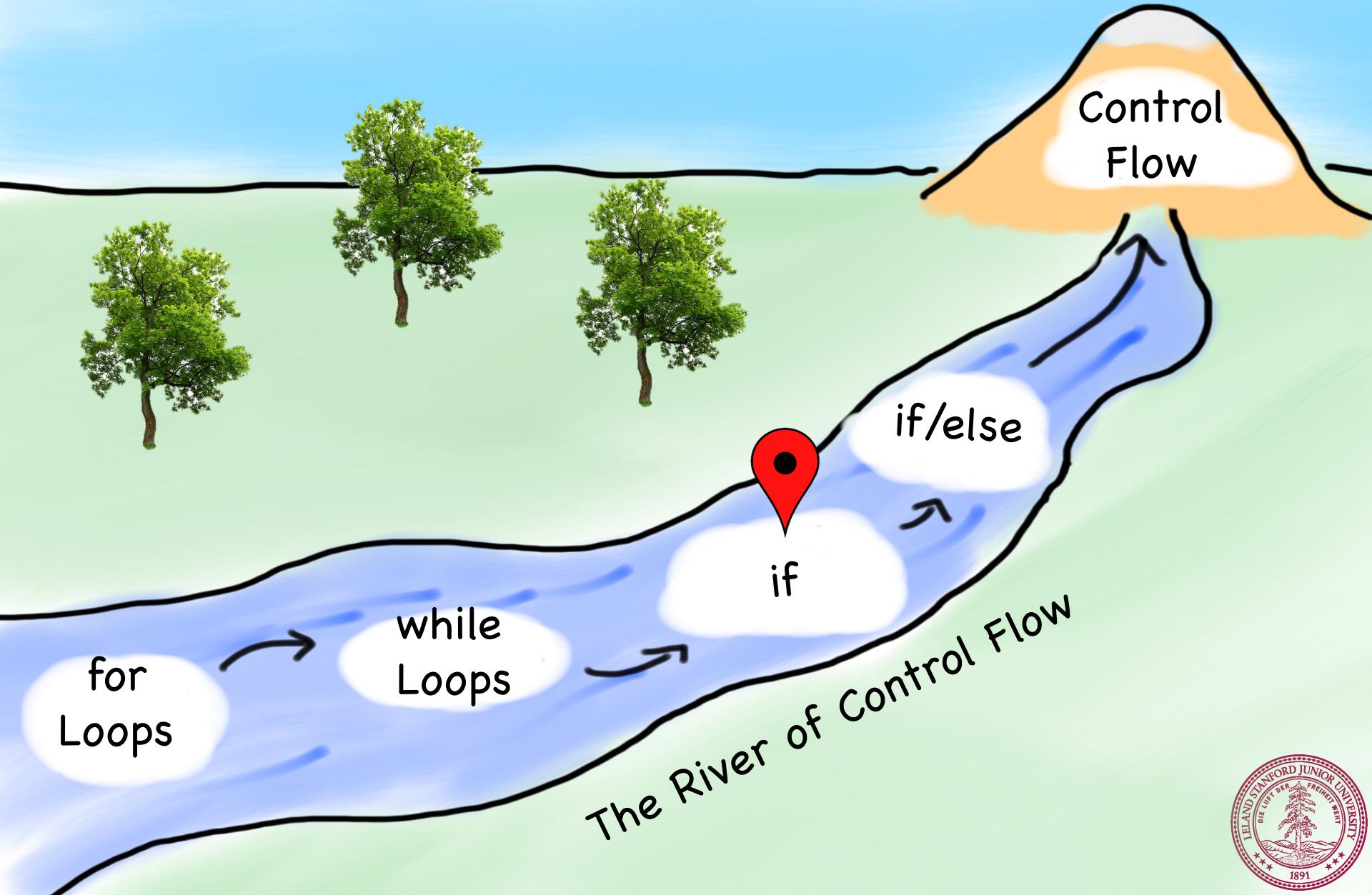
1700 closed down.



# Grace Hopper



# Today's Route



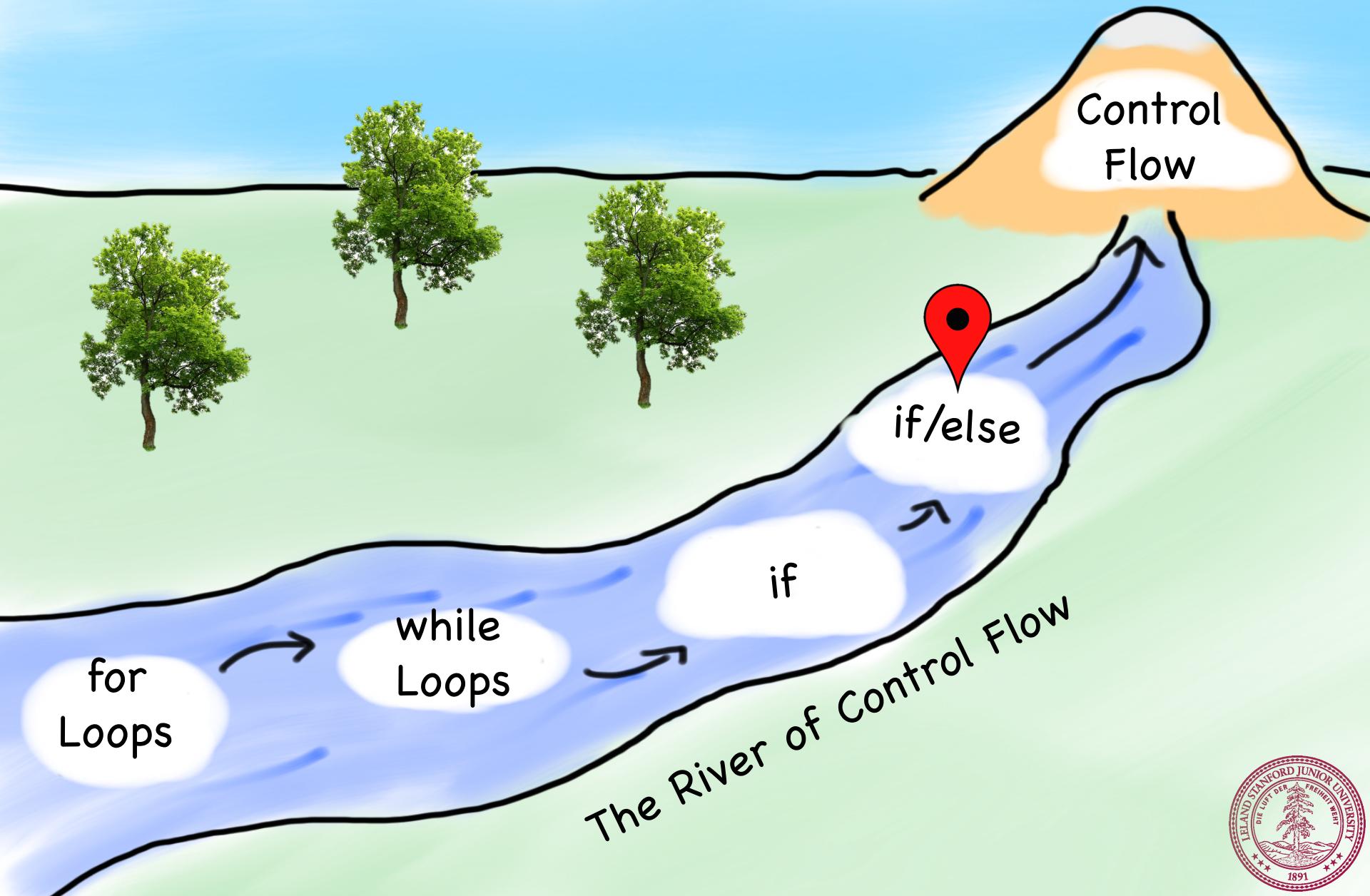
# if statement

```
if condition:
    statements                                # note indenting
```

---

```
def safe_pick_up():
    if beepers_present():
        pick_beeper() # note indenting
```

# Today's Route



# if-else statement

```
if condition:
    statements          # note indenting
else:
    statements          # note indenting
```

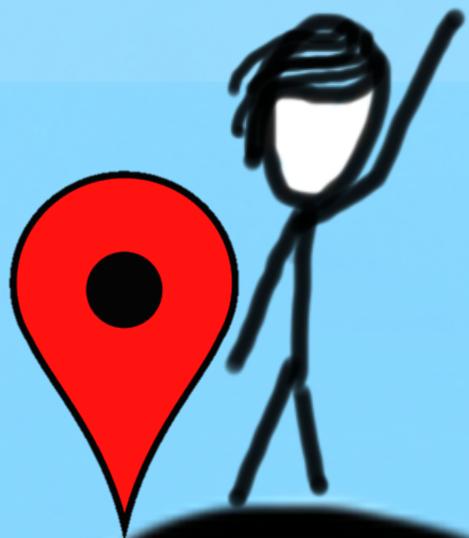
---

```
def invert_beeper():
    if beepers_present():
        pick_beeper() # note indenting
    else:
        put_beeper()  # note indenting
```

You just learned most of  
programming “control flow”

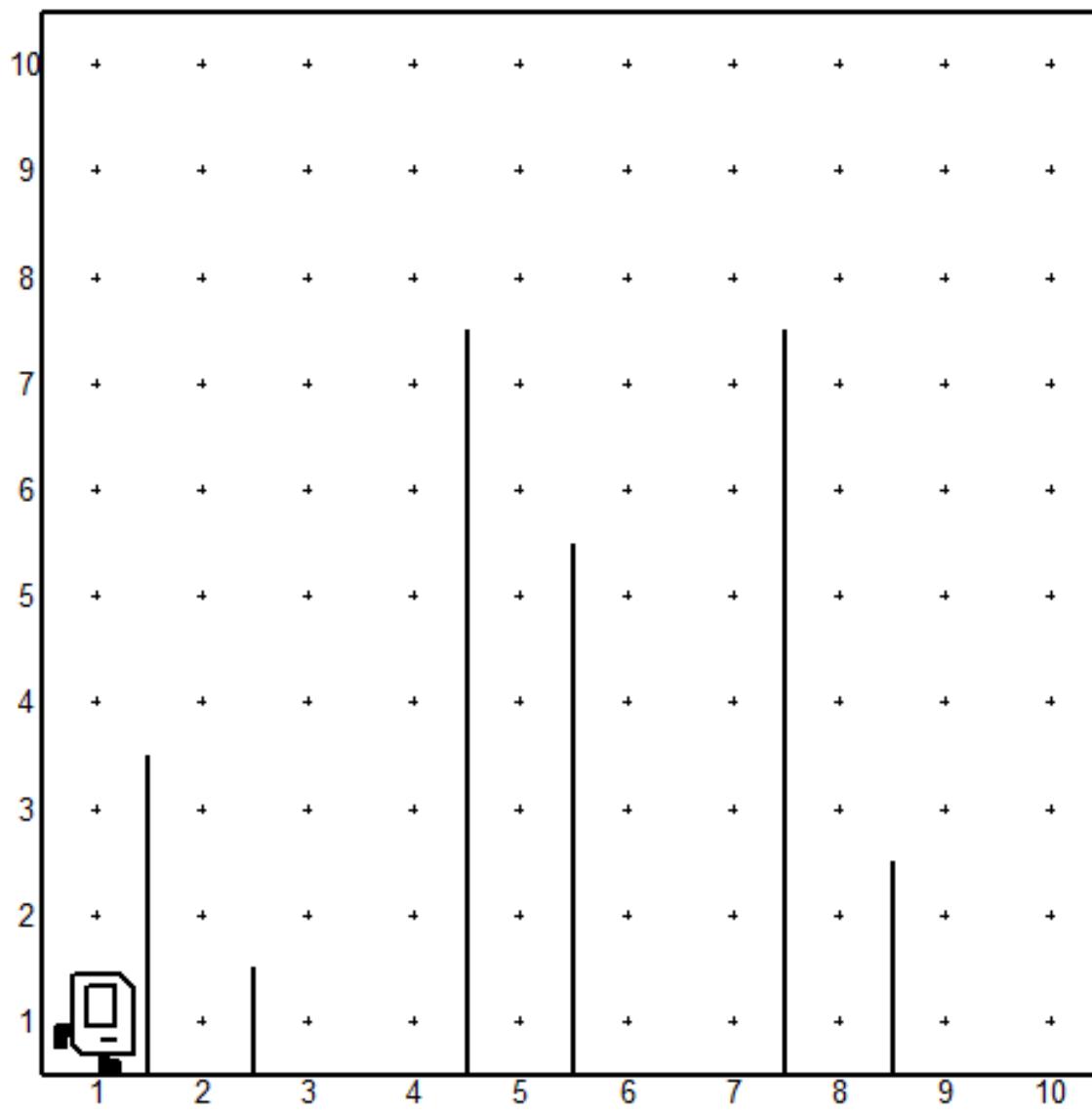
# Today's Goal

1. Code using loops and conditions
2. Trace programs that use loops and conditions

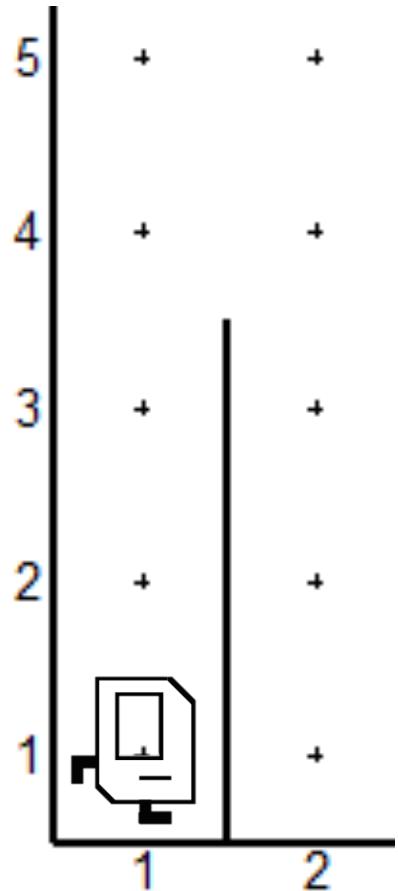


Putting it all together  
SteepChaseKarel.py

# Steeple Chase

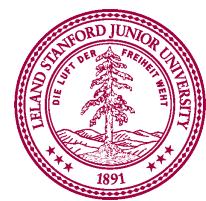
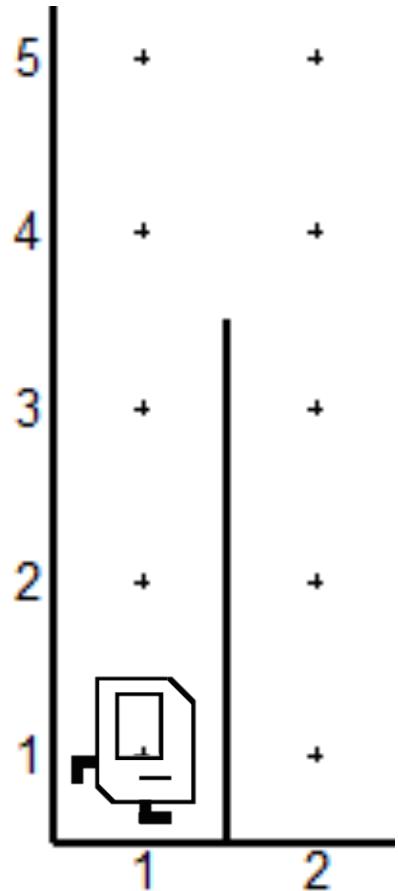


# Focus on One Steeple



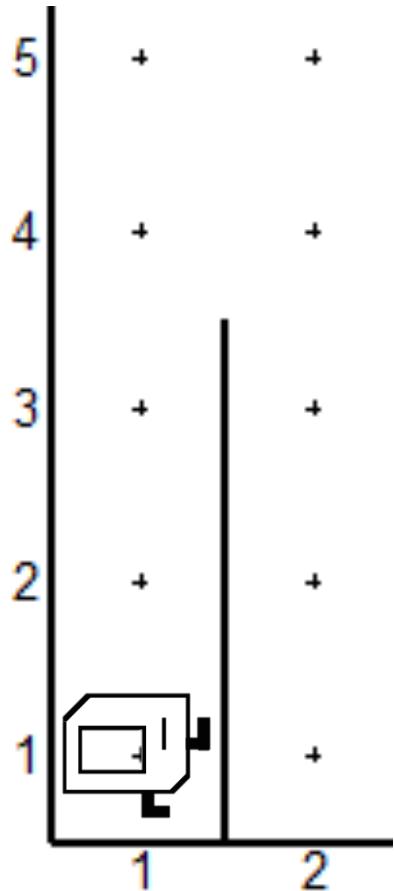
# Focus on One Steeple

`turn_left()`



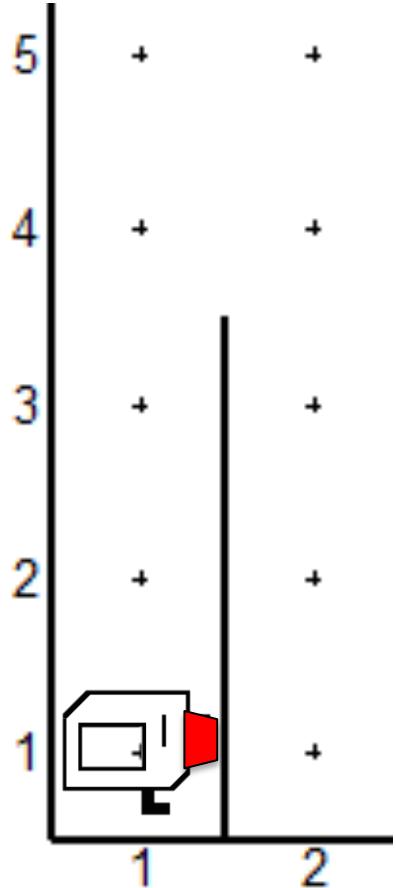
# Focus on One Steeple

turn\_left()



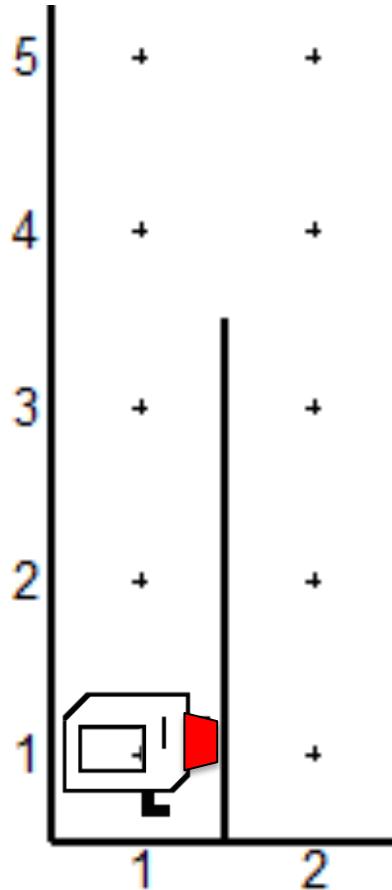
# Focus on One Steeple

```
turn_left()  
while right_is_blocked():  
    move()
```



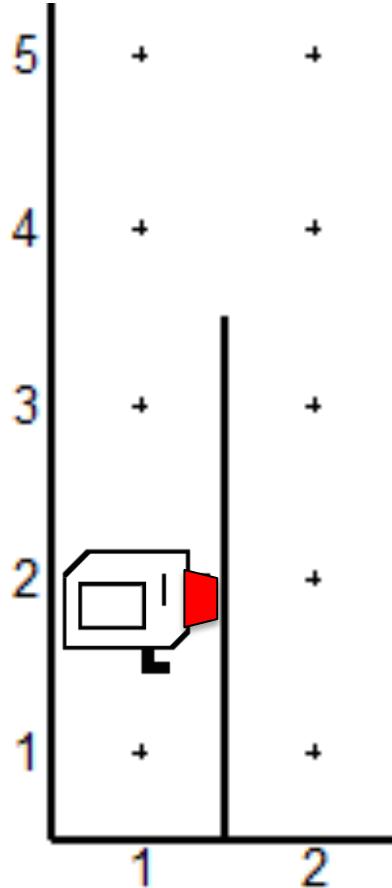
# Focus on One Steeple

```
turn_left()  
while right_is_blocked():  
    move()
```



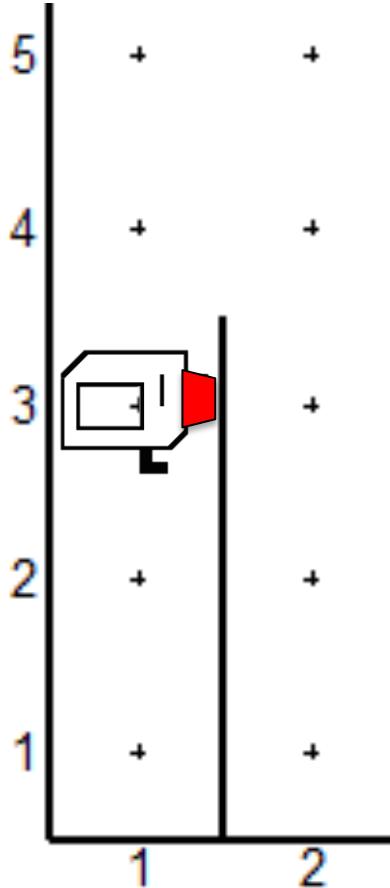
# Focus on One Steeple

```
turn_left()  
while right_is_blocked():  
    move()
```



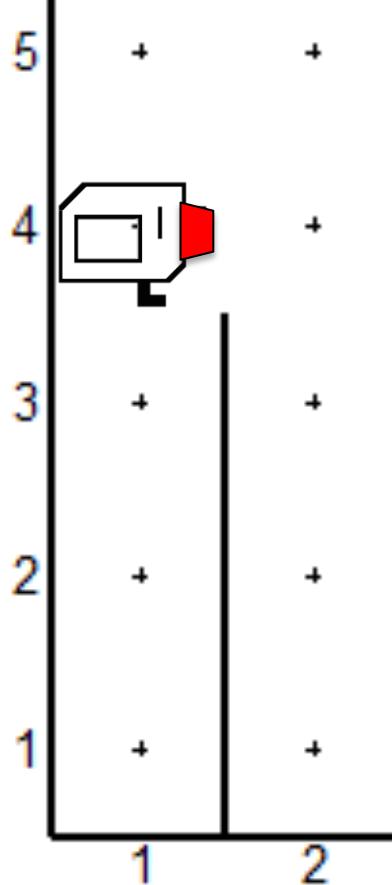
# Focus on One Steeple

```
turn_left()  
while right_is_blocked():  
    move()
```



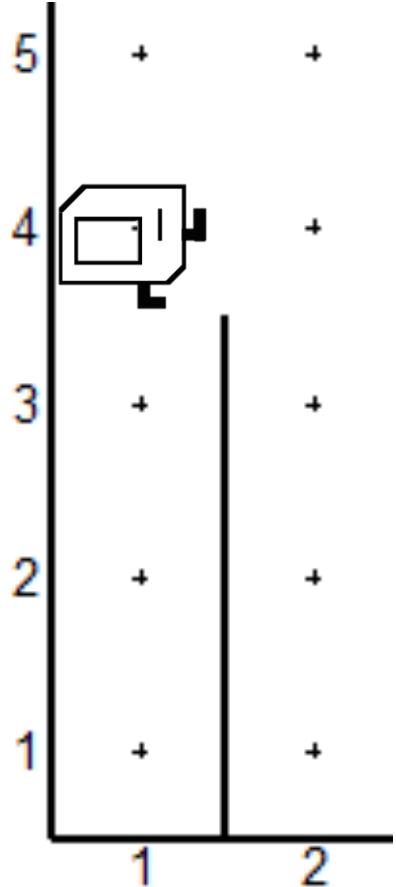
# Focus on One Steeple

```
turn_left()  
while right_is_blocked():  
    move()
```

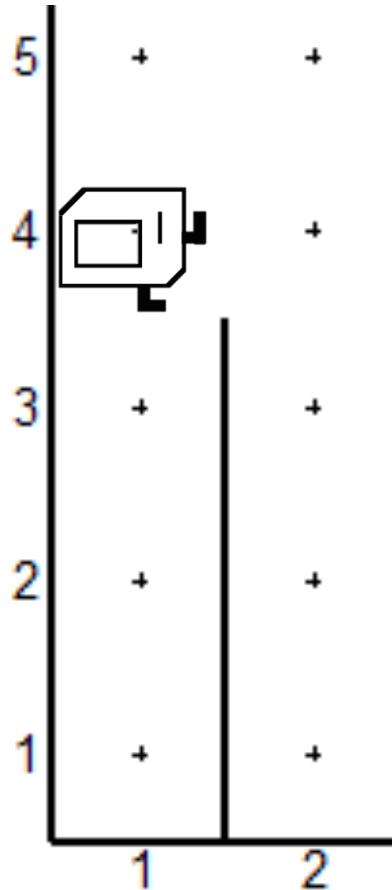


# Focus on One Steeple

```
turn_left()  
while right_is_blocked():  
    move()
```



# Focus on One Steeple

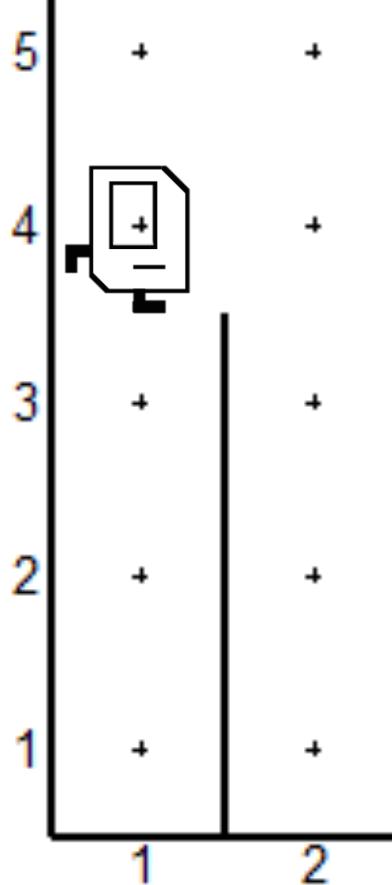


```
turn_left()  
while right_is_blocked():  
    move()  
turn_right()
```

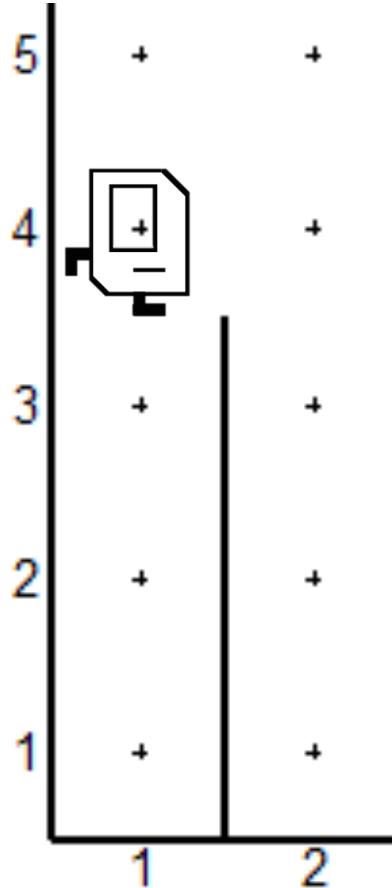


# Focus on One Steeple

```
turn_left()  
while right_is_blocked():  
    move()  
    turn_right()
```



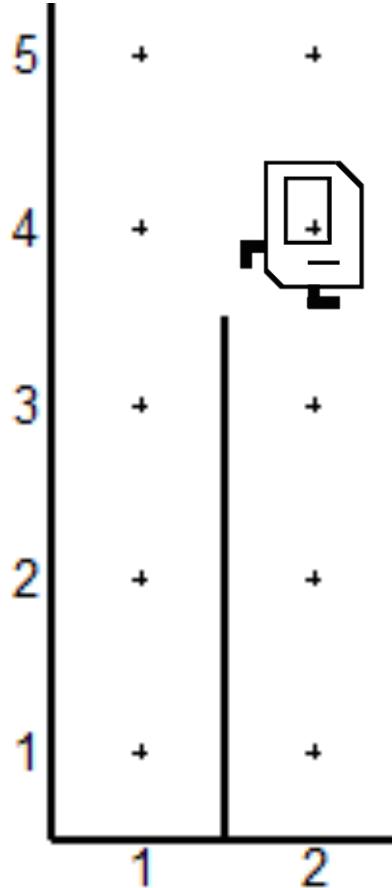
# Focus on One Steeple



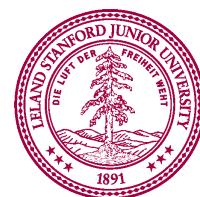
```
turn_left()  
while right_is_blocked():  
    move()  
turn_right()  
move()
```



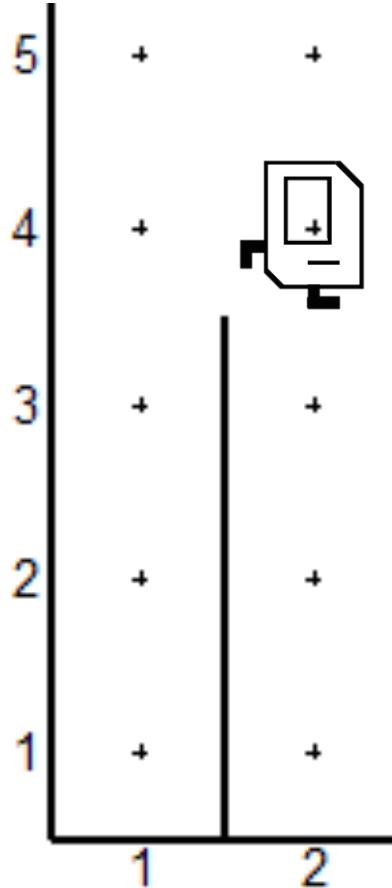
# Focus on One Steeple



```
turn_left()  
while right_is_blocked():  
    move()  
turn_right()  
move()
```



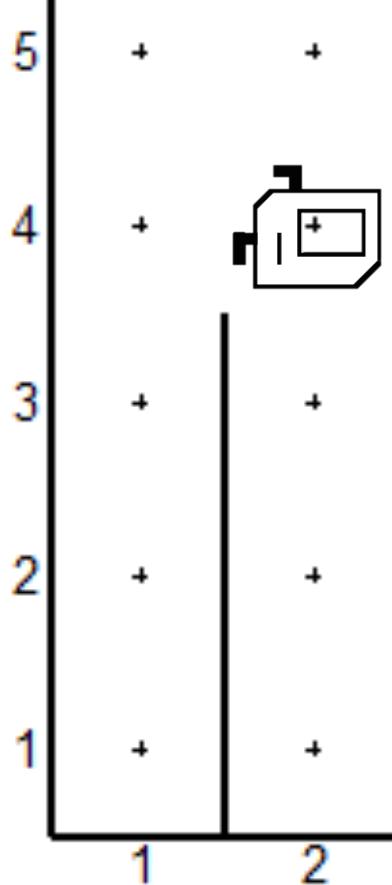
# Focus on One Steeple



```
turn_left()  
while right_is_blocked():  
    move()  
    turn_right()  
move()  
turn_right()
```



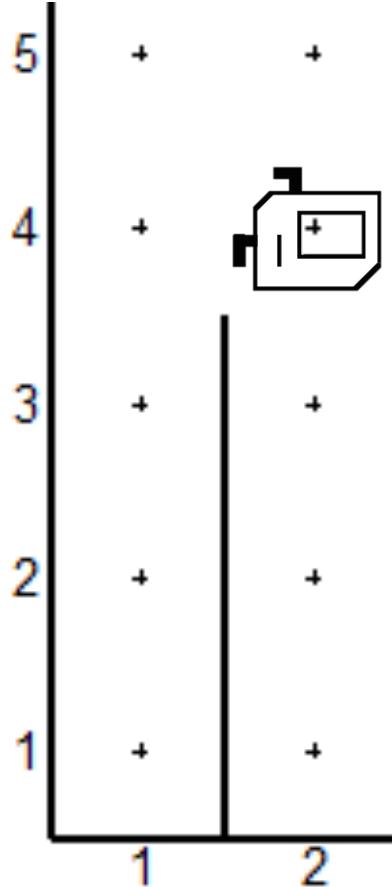
# Focus on One Steeple



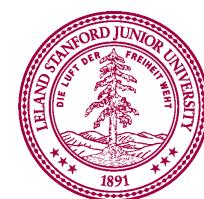
```
turn_left()  
while right_is_blocked():  
    move()  
    turn_right()  
    move()  
    turn_right()
```



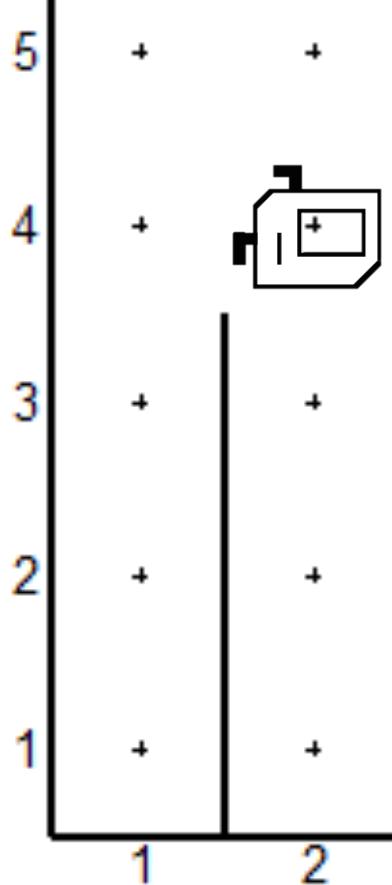
# Focus on One Steeple



```
turn_left()  
while right_is_blocked():  
    move()  
    turn_right()  
    move()  
    turn_right()  
    move_to_wall()
```



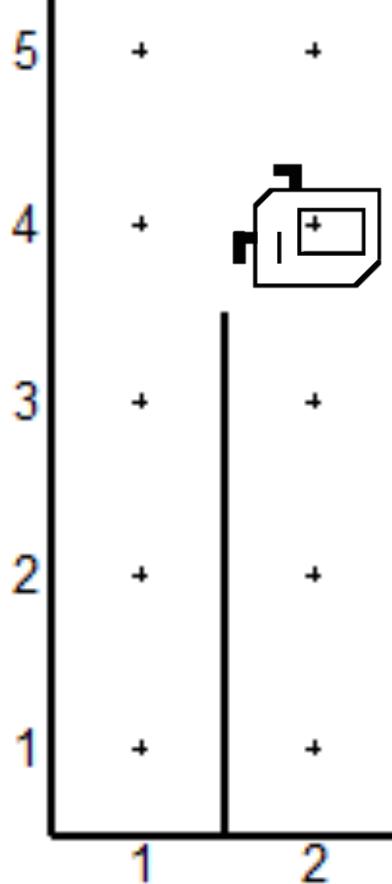
# Focus on One Steeple



```
turn_left()  
while right_is_blocked():  
    move()  
turn_right()  
move()  
turn_right()  
move_to_wall()  
  
def move_to_wall():  
    while front_is_clear():  
        move()
```

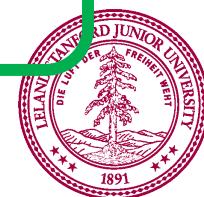


# Focus on One Steeple

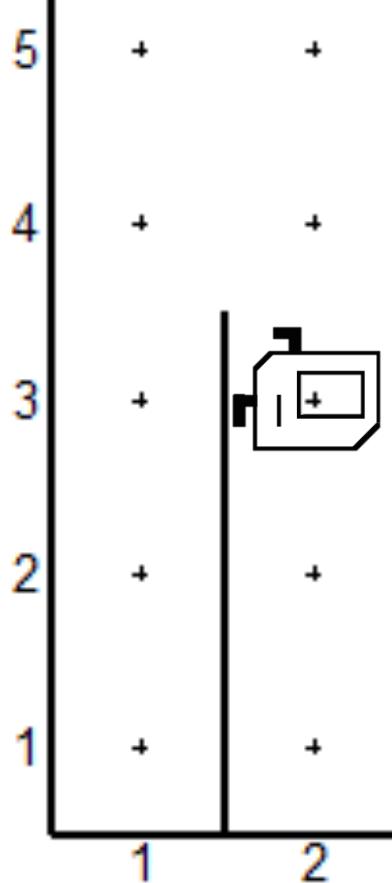


```
turn_left()  
while right_is_blocked():  
    move()  
turn_right()  
move()  
turn_right()  
move_to_wall()
```

```
def move_to_wall():  
    while front_is_clear():  
        move()
```

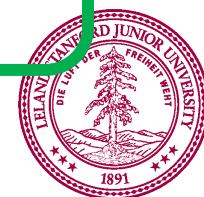


# Focus on One Steeple

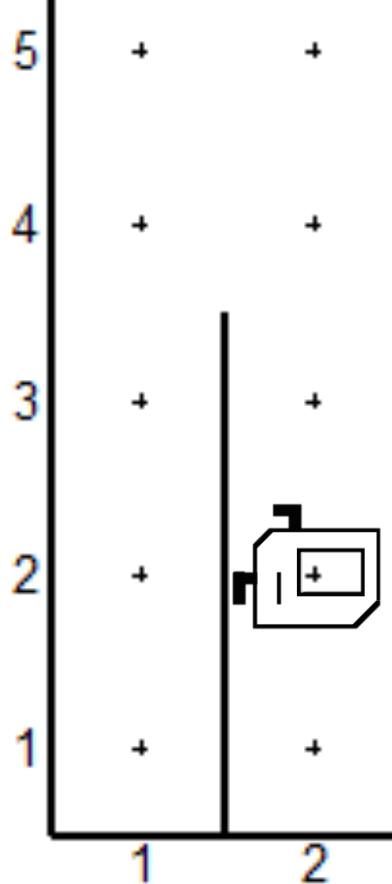


```
turn_left()  
while right_is_blocked():  
    move()  
    turn_right()  
    move()  
    turn_right()  
    move_to_wall()
```

```
def move_to_wall():  
    while front_is_clear():  
        move()
```

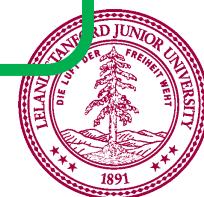


# Focus on One Steeple

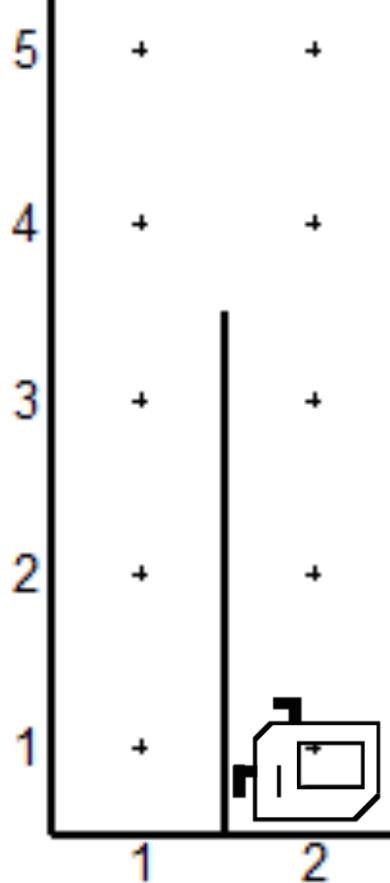


```
turn_left()  
while right_is_blocked():  
    move()  
    turn_right()  
    move()  
    turn_right()  
    move_to_wall()
```

```
def move_to_wall():  
    while front_is_clear():  
        move()
```

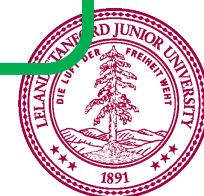


# Focus on One Steeple

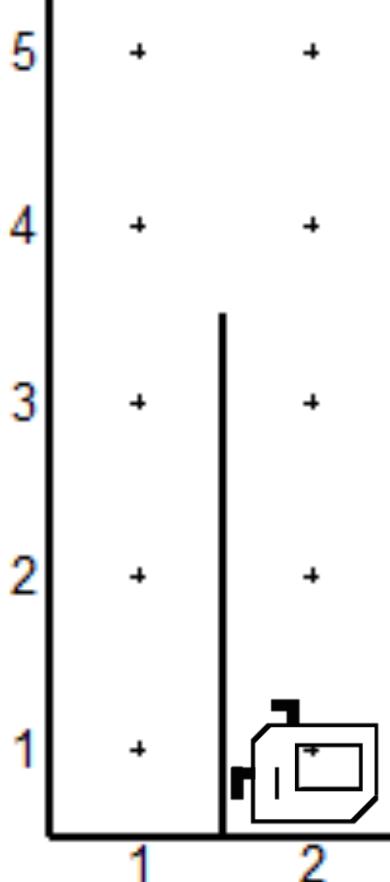


```
turn_left()  
while right_is_blocked():  
    move()  
    turn_right()  
    move()  
    turn_right()  
    move_to_wall()
```

```
def move_to_wall():  
    while front_is_clear():  
        move()
```



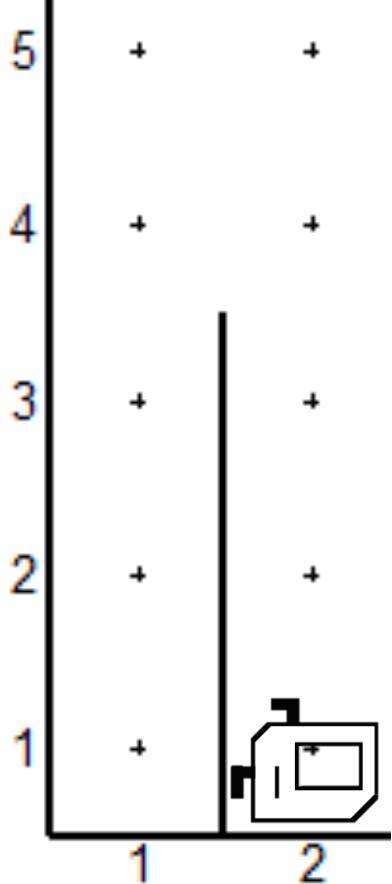
# Focus on One Steeple



```
turn_left()  
while right_is_blocked():  
    move()  
  
turn_right()  
move()  
turn_right()  
move_to_wall()  
  
def move_to_wall():  
    while front_is_clear():  
        move()
```



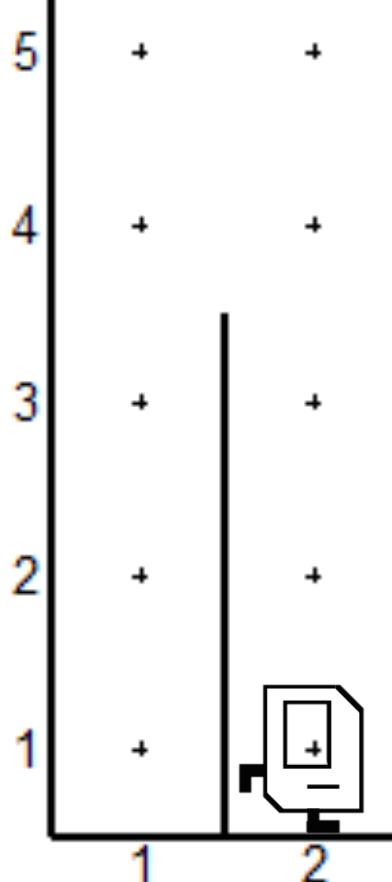
# Focus on One Steeple



```
turn_left()  
while right_is_blocked():  
    move()  
    turn_right()  
    move()  
    turn_right()  
    move_to_wall()  
    turn_left()  
  
def move_to_wall():  
    while front_is_clear():  
        move()
```



# Focus on One Steeple

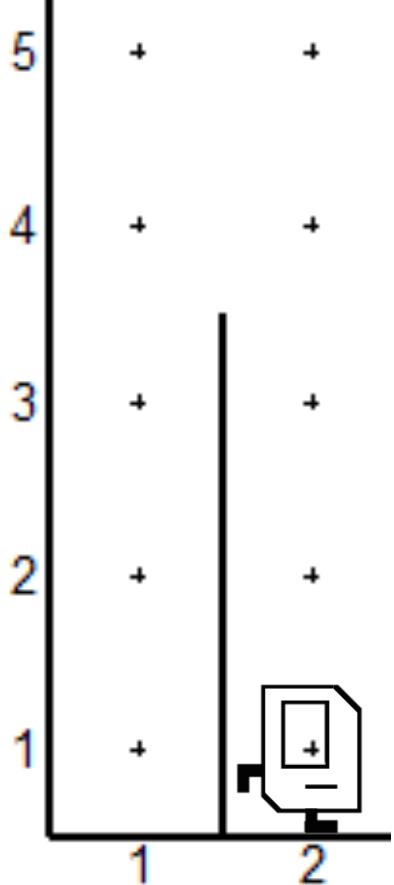


```
turn_left()  
while right_is_blocked():  
    move()  
    turn_right()  
    move()  
    turn_right()  
    move_to_wall()  
    turn_left()
```

```
def move_to_wall():  
    while front_is_clear():  
        move()
```



# Focus on One Steeple



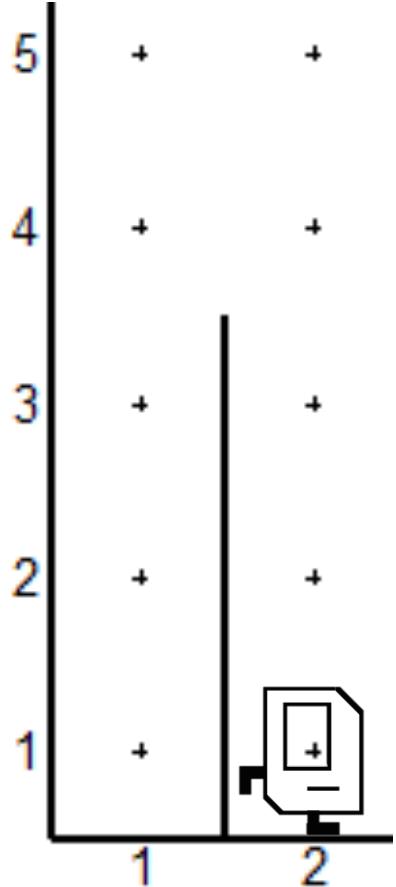
```
turn_left()  
while right_is_blocked():  
    move()  
turn_right()  
move()  
turn_right()  
move_to_wall()  
turn_left()
```

You need the **postcondition** of a loop to match the **precondition**

```
def move_to_wall():  
    while front_is_clear():  
        move()
```



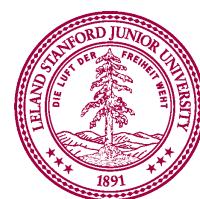
# Focus on One Steeple



```
turn_left()  
while right_is_blocked():  
    move()  
    turn_right()  
    move()  
    turn_right()  
    move_to_wall()  
    turn_left()
```

ascend\_hurdle()

descend\_hurdle()



# Focus on One Steeple

```
turn_left()  
while right_is_blocked():  
    move()  
    turn_right()  
    move()  
    turn_right()  
    move_to_wall()  
    turn_left()
```

ascend\_hurdle()

descend\_hurdle()



# Focus on One Steeple

```
def ascend_hurdle():
    turn_left()
    while right_is_blocked():
        move()
    turn_right()
```

```
ascend_hurdle()
move()
turn_right()
move_to_wall()
turn_left()
```

descend\_hurdle()

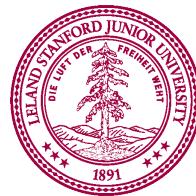


# Focus on One Steeple

```
def ascend_hurdle():
    turn_left()
    while right_is_blocked():
        move()
    turn_right()

def descend_hurdle():
    turn_right()
    move_to_wall()
    turn_left()
```

ascend\_hurdle()  
move()  
descend\_hurdle()



# Focus on One Steeple

```
def ascend_hurdle():
    turn_left()
    while right_is_blocked():
        move()
    turn_right()

def descend_hurdle():
    turn_right()
    move_to_wall()
    turn_left()

def jump_hurdle():
    ascend_hurdle()
    move()
    descend_hurdle()
```



A Whole Program:  
SteepChaseKarel.py