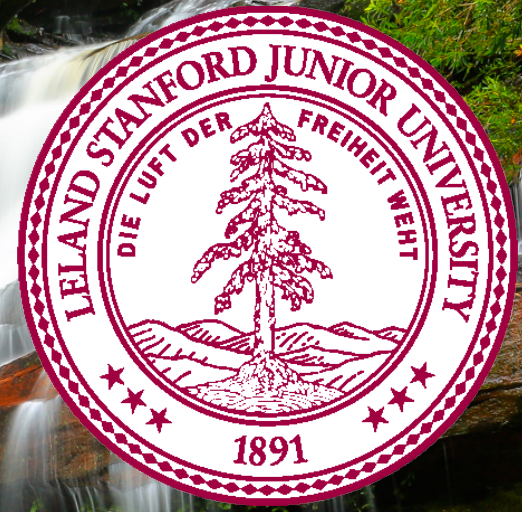
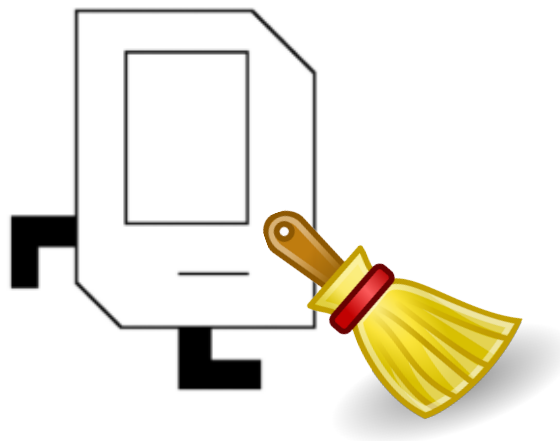




Control Flow

Chris Piech and Mehran Sahami
CS106A, Stanford University

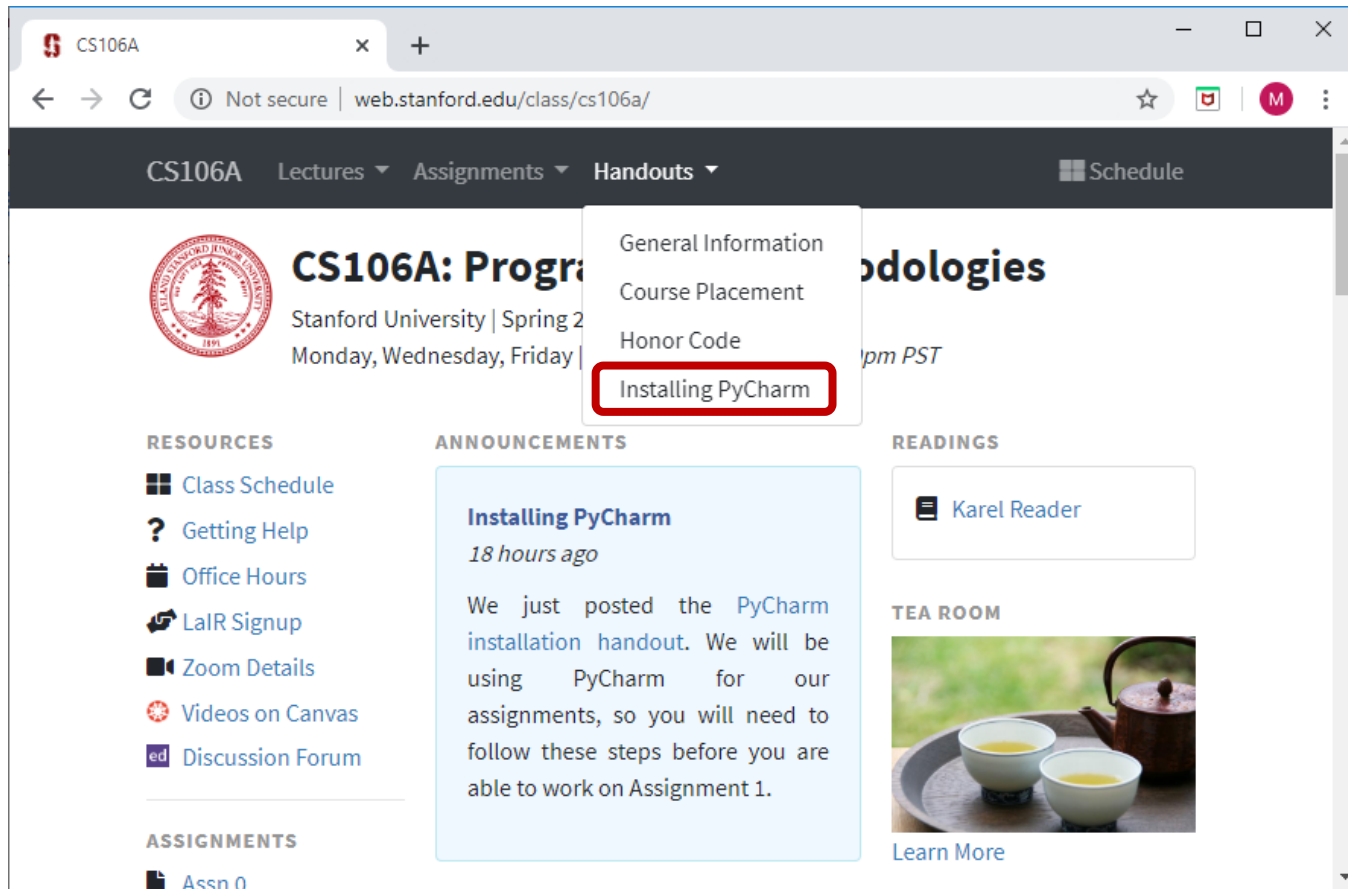
Housekeeping



- Class website: <http://cs106a.stanford.edu>
- Section sign-ups
 - Sign-up at: <http://cs198.stanford.edu> (will be on cs106a page)
 - Sign-ups start Thursday, April 9 at 5pm (PDT)
 - Sign-ups end Sunday, April 12 at 5pm (PDT)
 - Not first-come, first-served, but make sure to sign-up!

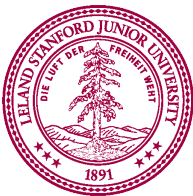


Install PyCharm

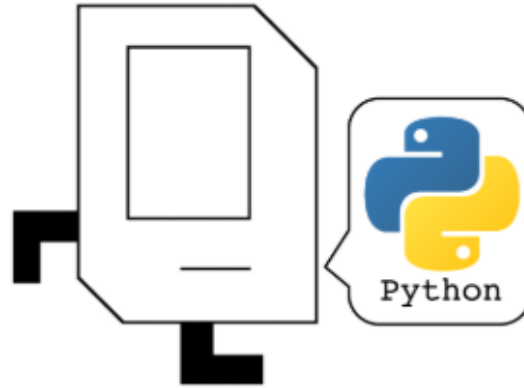


The screenshot shows a web browser window displaying the CS106A course page on Stanford University's website. The browser's address bar shows the URL `web.stanford.edu/class/cs106a/`. The page has a dark navigation bar with links for "Lectures", "Assignments", "Handouts", and "Schedule". Below the navigation bar, the page features the Stanford University seal, the course title "CS106A: Program", and the semester "Stanford University | Spring 2020". A dropdown menu is open, showing options: "General Information", "Course Placement", "Honor Code", and "Installing PyCharm", with the last option highlighted by a red rectangle. The main content area is divided into three columns: "RESOURCES" (Class Schedule, Getting Help, Office Hours, LaIR Signup, Zoom Details, Videos on Canvas, Discussion Forum), "ANNOUNCEMENTS" (a post titled "Installing PyCharm" from 18 hours ago), and "READINGS" (Karel Reader). A "TEA ROOM" section with a photo of tea is also visible.

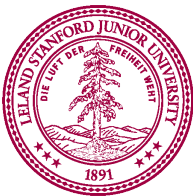
Please follow instructions *closely*.
Post on Ed if you have problems.



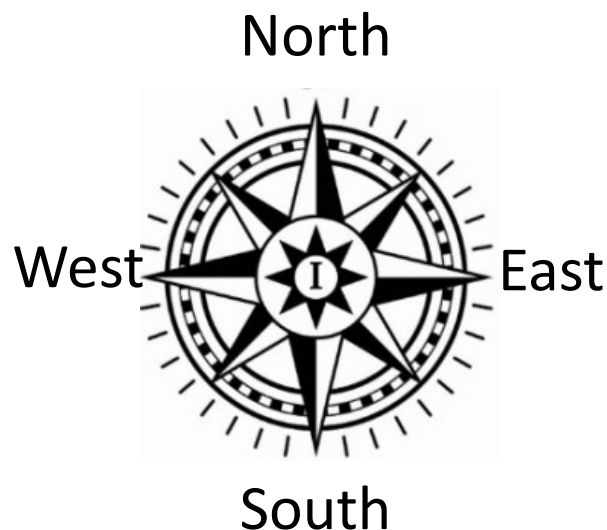
Using Karel and Assignment 1



- 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 1:30pm (PDT) on Friday, April 17th
- 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.
```

```
"""
```

Multi-line
comment

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

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

One line
comment

```
# Karel turns to the right
```

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

Descriptive
names
(snake_case)

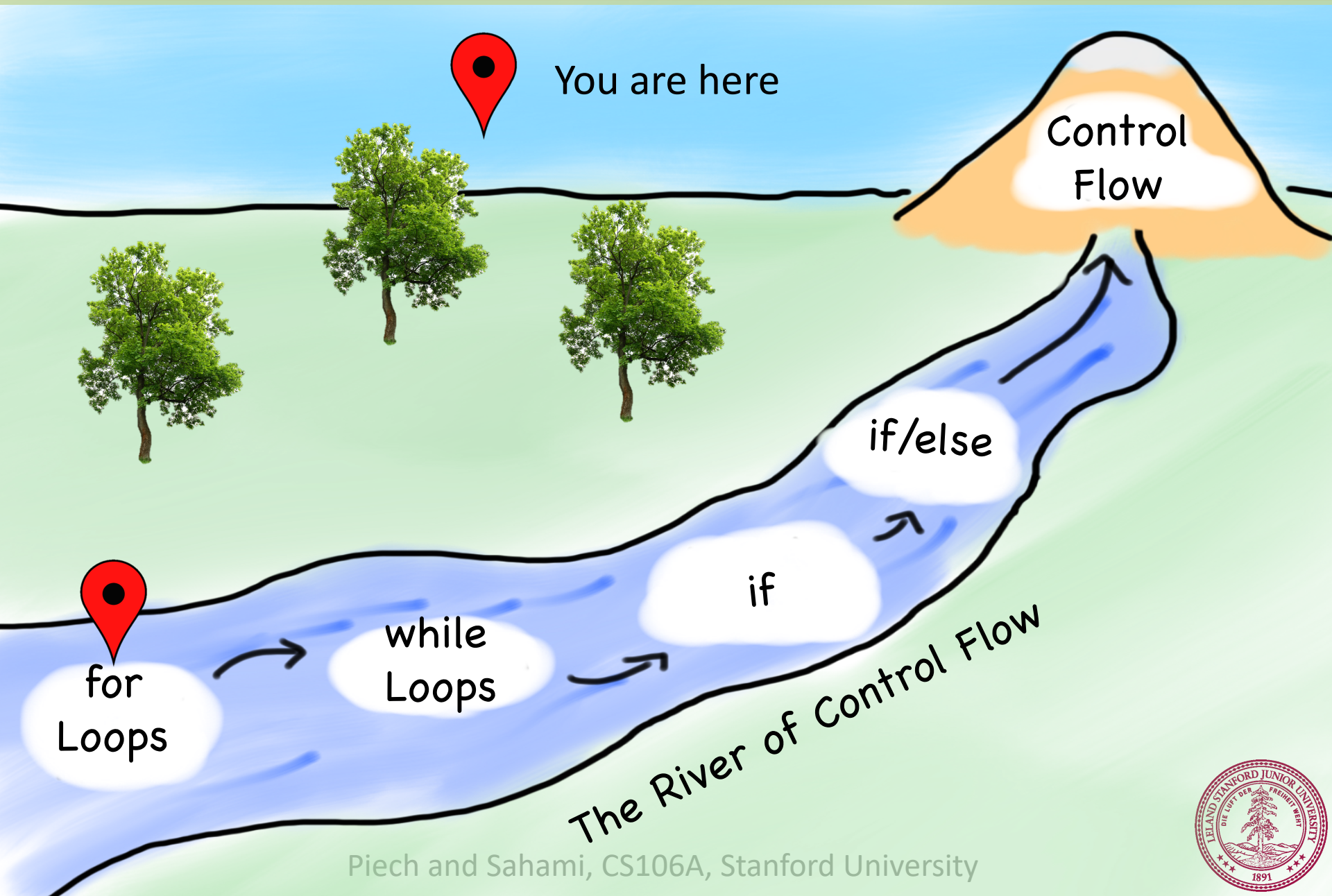


Today's Goal

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
```

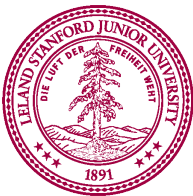
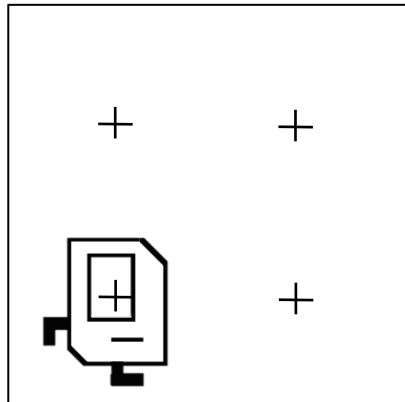


```
def turn_right():  
    for i in range(3):  
        turn_left()          # note indenting
```



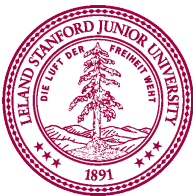
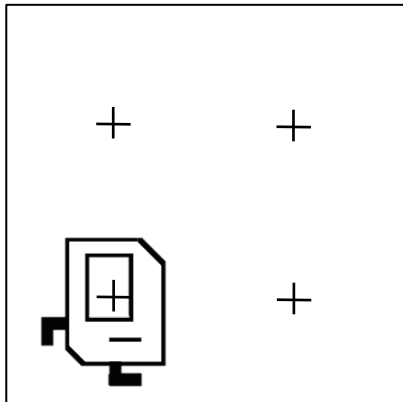
Place Beeper Square

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



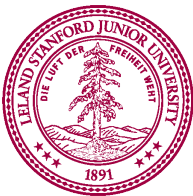
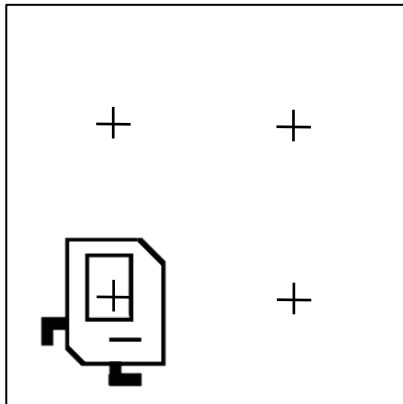
Place Beeper Square

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



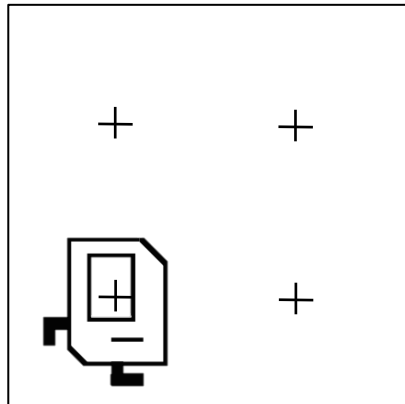
Place Beeper Square

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

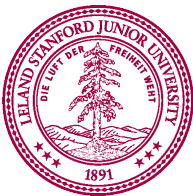


Place Beeper Square

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

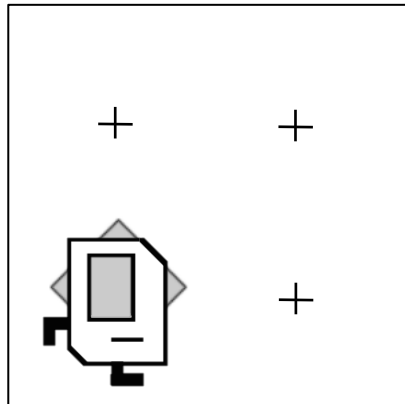


First time
through the
loop

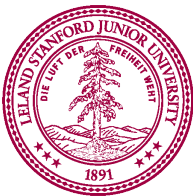


Place Beeper Square

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

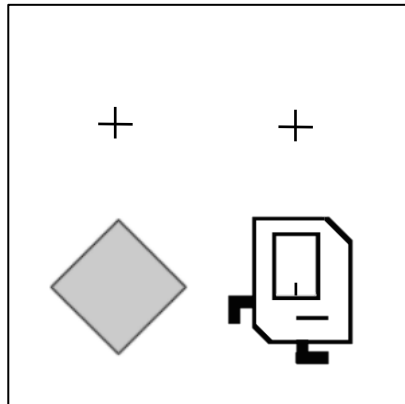


First time
through the
loop

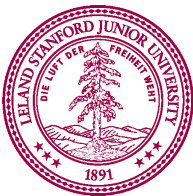


Place Beeper Square

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

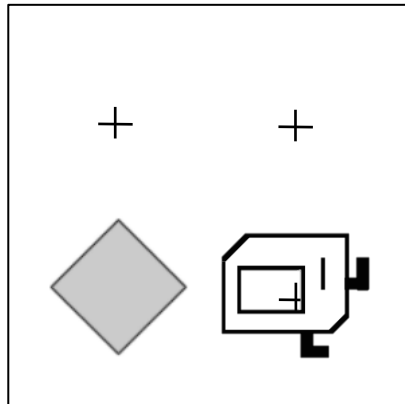


First time
through the
loop

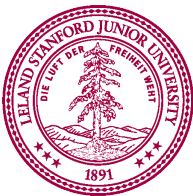


Place Beeper Square

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

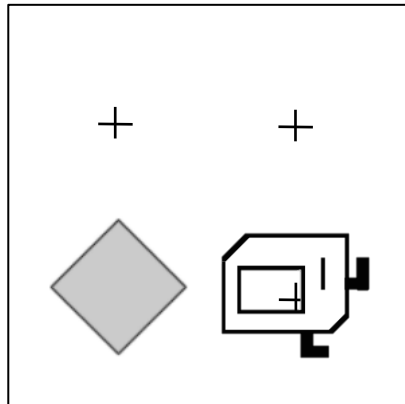


First time
through the
loop

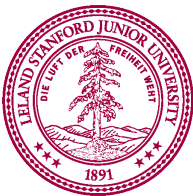


Place Beeper Square

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

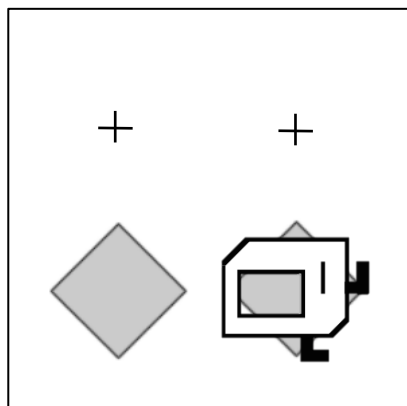


Second time
through the
loop



Place Beeper Square

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

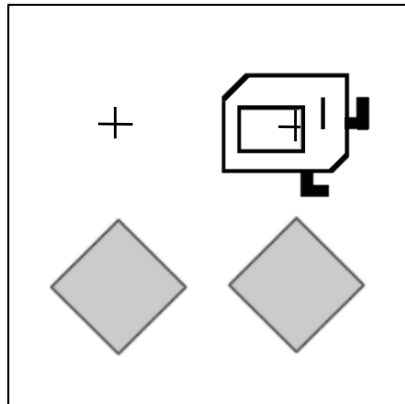


Second time
through the
loop



Place Beeper Square

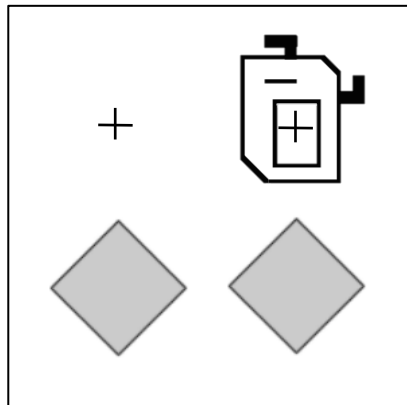
```
def main():  
    for i in range(4):  
        put_beeper()  
        move()  
        turn_left()
```



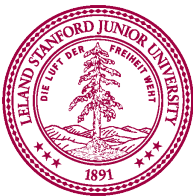
Second time
through the
loop

Place Beeper Square

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

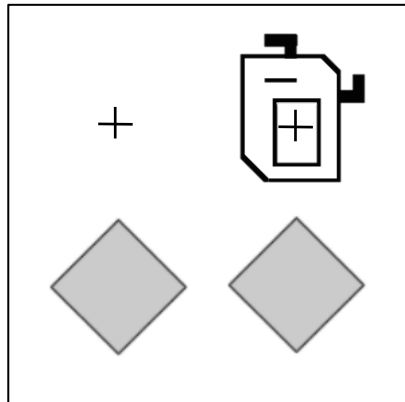


Second time
through the
loop

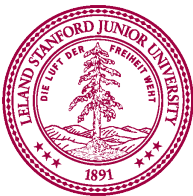


Place Beeper Square

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

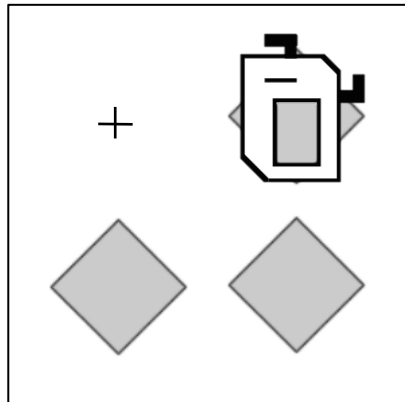


Third time
through the
loop

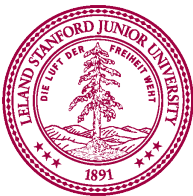


Place Beeper Square

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

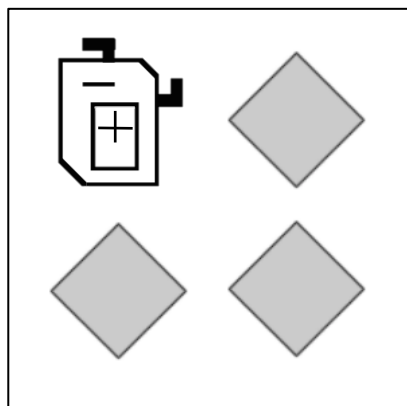


Third time
through the
loop



Place Beeper Square

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

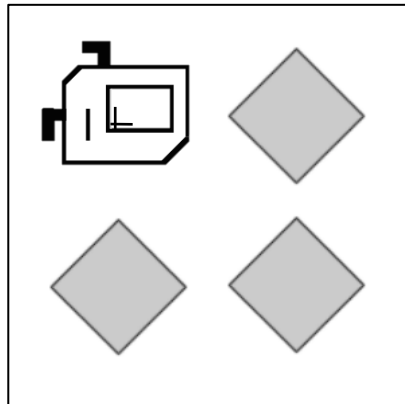


Third time
through the
loop

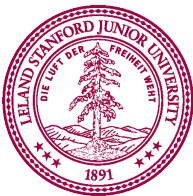


Place Beeper Square

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

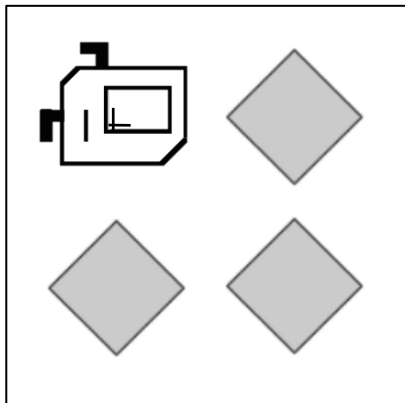


Third time
through the
loop



Place Beeper Square

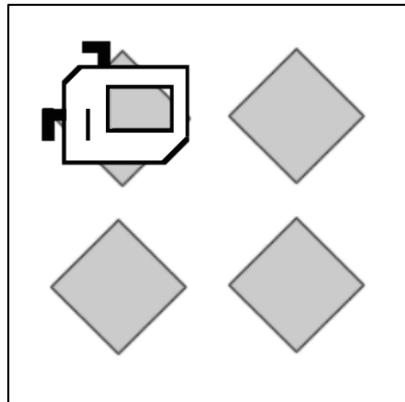
```
def main():  
    for i in range(4):  
        put_beeper()  
        move()  
        turn_left()
```



Fourth time
through the
loop

Place Beeper Square

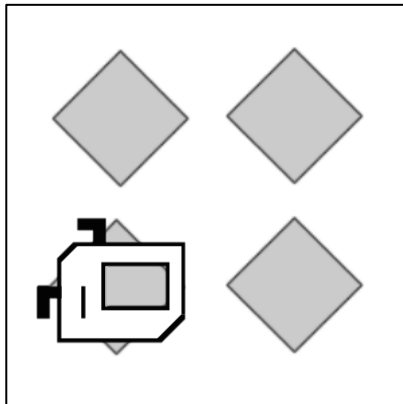
```
def main():  
    for i in range(4):  
        put_beeper()  
        move()  
        turn_left()
```



Fourth time
through the
loop

Place Beeper Square

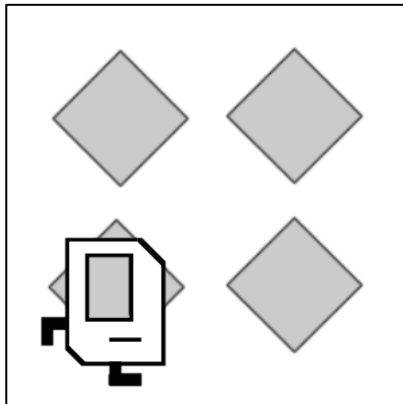
```
def main():  
    for i in range(4):  
        put_beeper()  
        move()  
        turn_left()
```



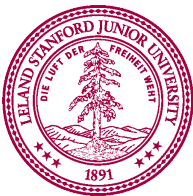
Fourth time
through the
loop

Place Beeper Square

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



Fourth time
through the
loop



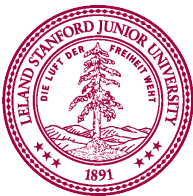
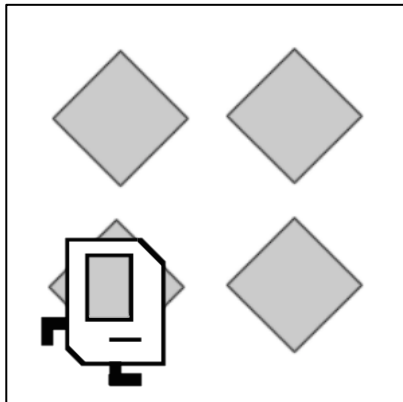
Place Beeper Square

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

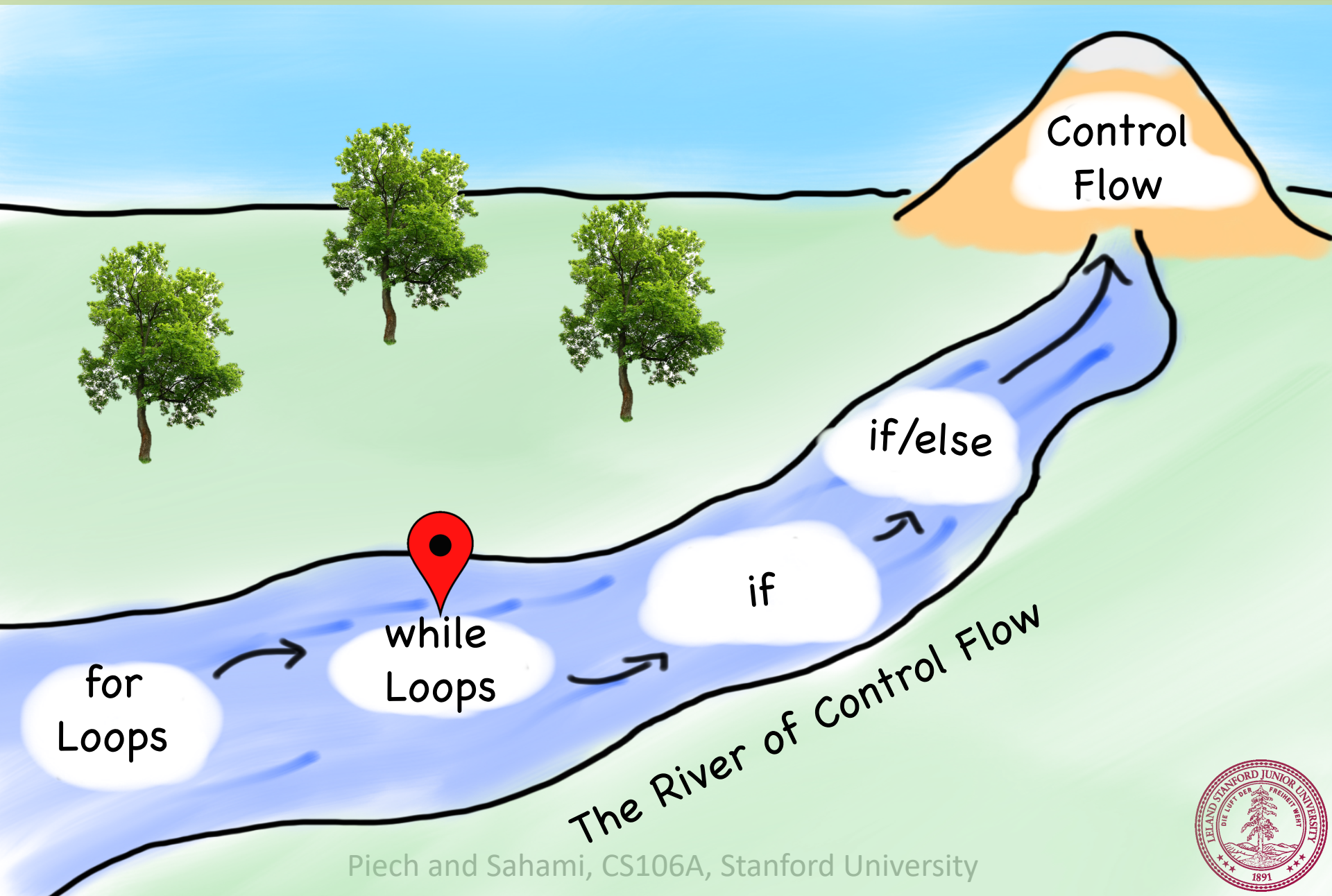


You need the **postcondition** of a loop to match the **precondition**

Done!



Today's Route



while loop

```
while condition:
```

```
    statements
```

```
# note indenting
```

```
def move_to_wall():
```

```
    while front_is_clear():
```

```
        move()
```

```
# note indenting
```

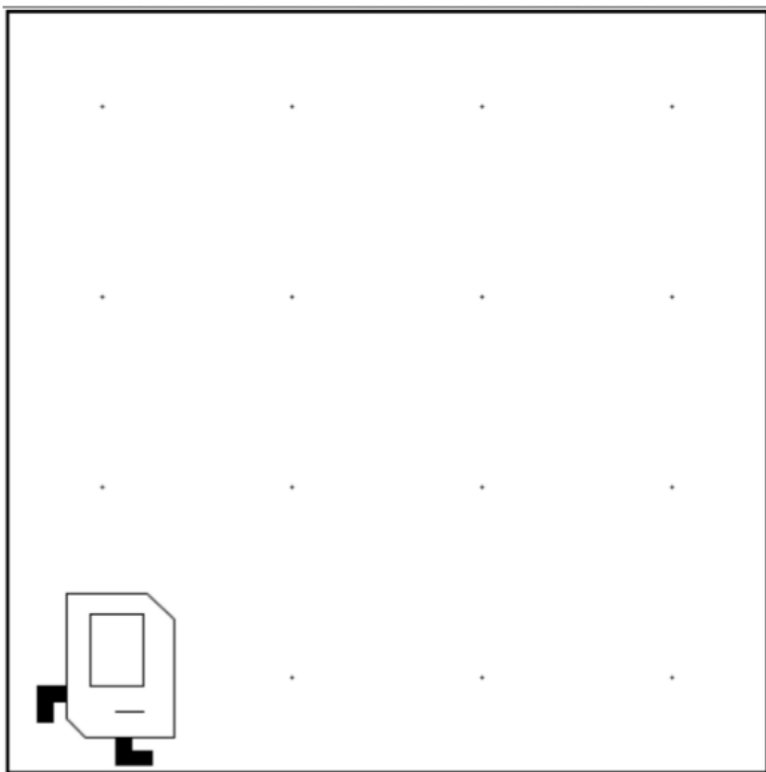
Conditions Karel Can Check For

<i>Test</i>	<i>Opposite</i>	<i>What it checks</i>
<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_beepers_present()</code>	Are there beepers on this corner?
<code>beepers_in_bag()</code>	<code>no_beepers_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?

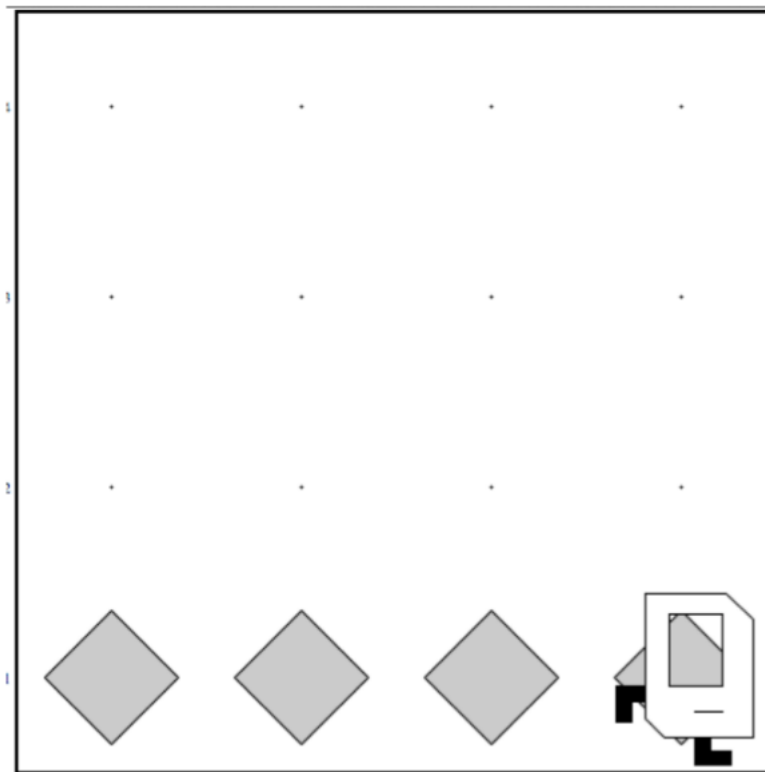
This is in Chapter 10 of the Karel course reader

Task: Place Beeper Line

Before

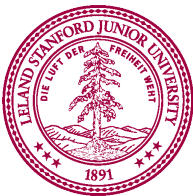


After



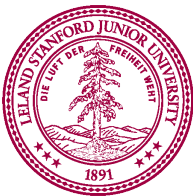
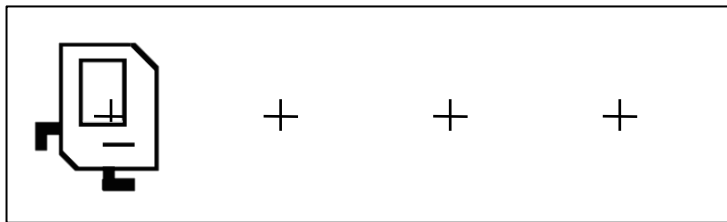
Place Beeper Line

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



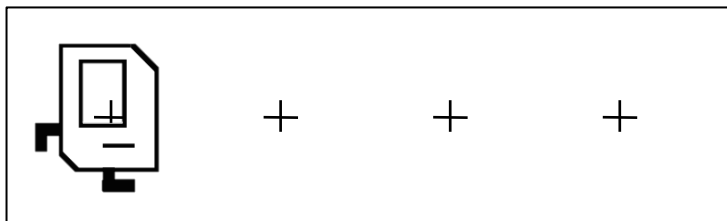
Place Beeper Line

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



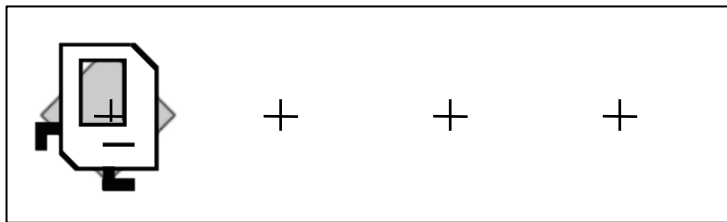
Place Beeper Line

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



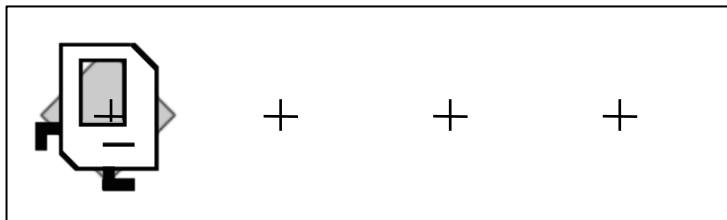
Place Beeper Line

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



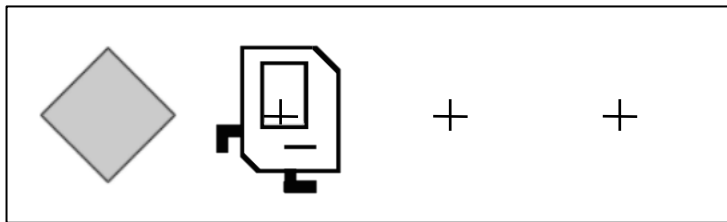
Place Beeper Line

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



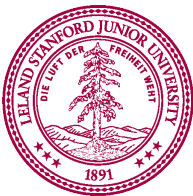
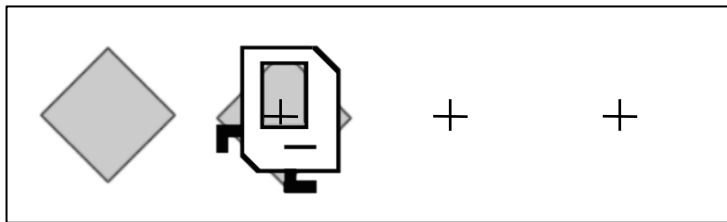
Place Beeper Line

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



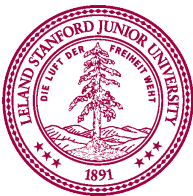
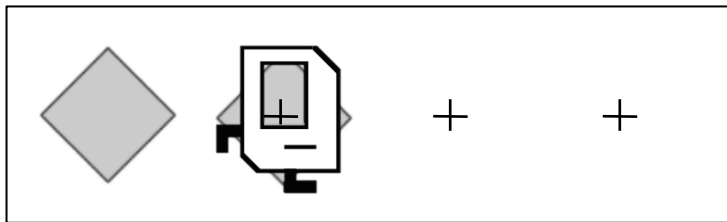
Place Beeper Line

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



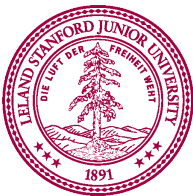
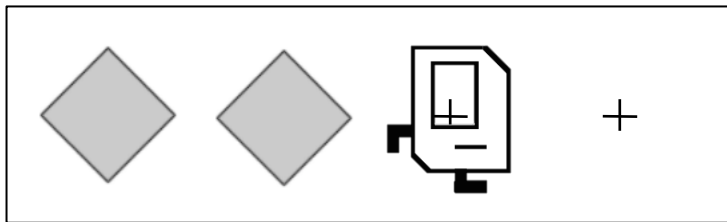
Place Beeper Line

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



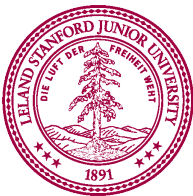
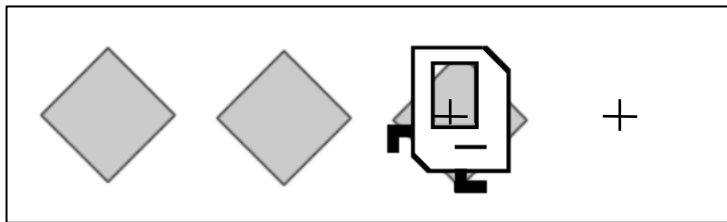
Place Beeper Line

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



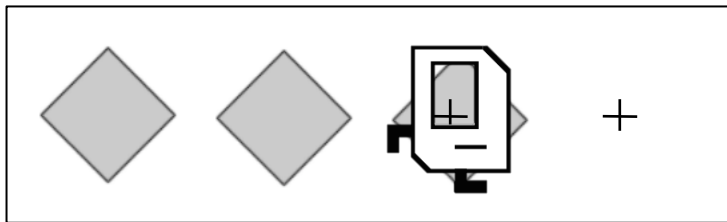
Place Beeper Line

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



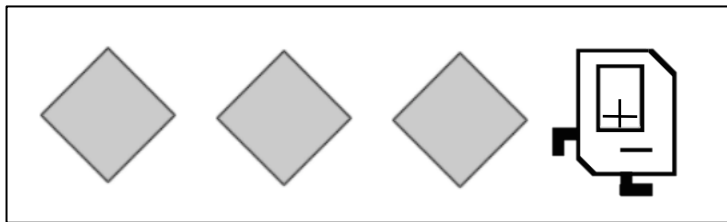
Place Beeper Line

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



Place Beeper Line

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



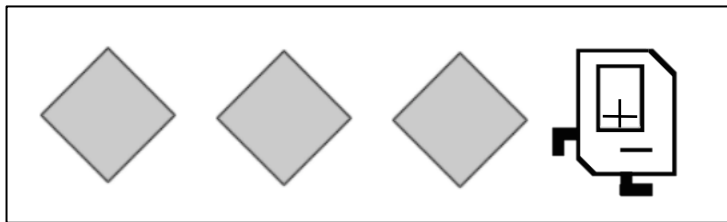
Place Beeper Line

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

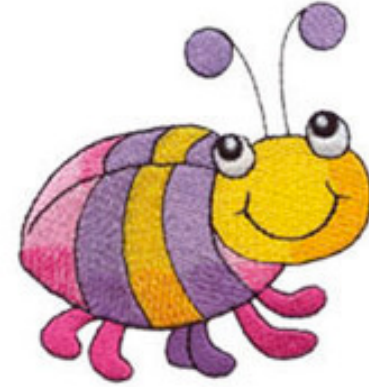


BUGGY!

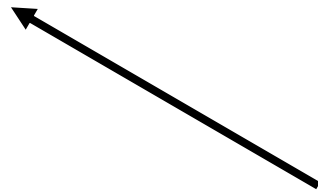
Done!



Place Beeper Line

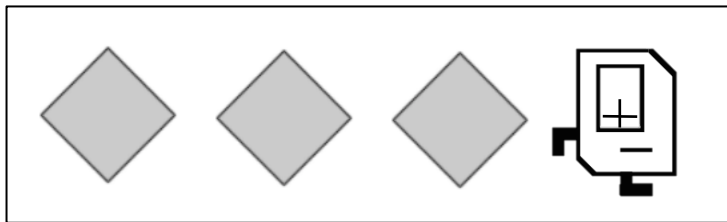


```
def main():  
    while front_is_clear():  
        put_beeper()  
        move()  
    put_beeper()           # add final put_beeper
```



Not in **while** loop

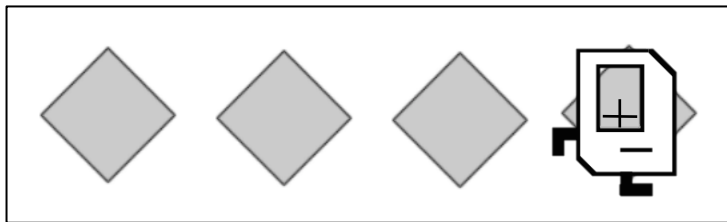
Fixed!



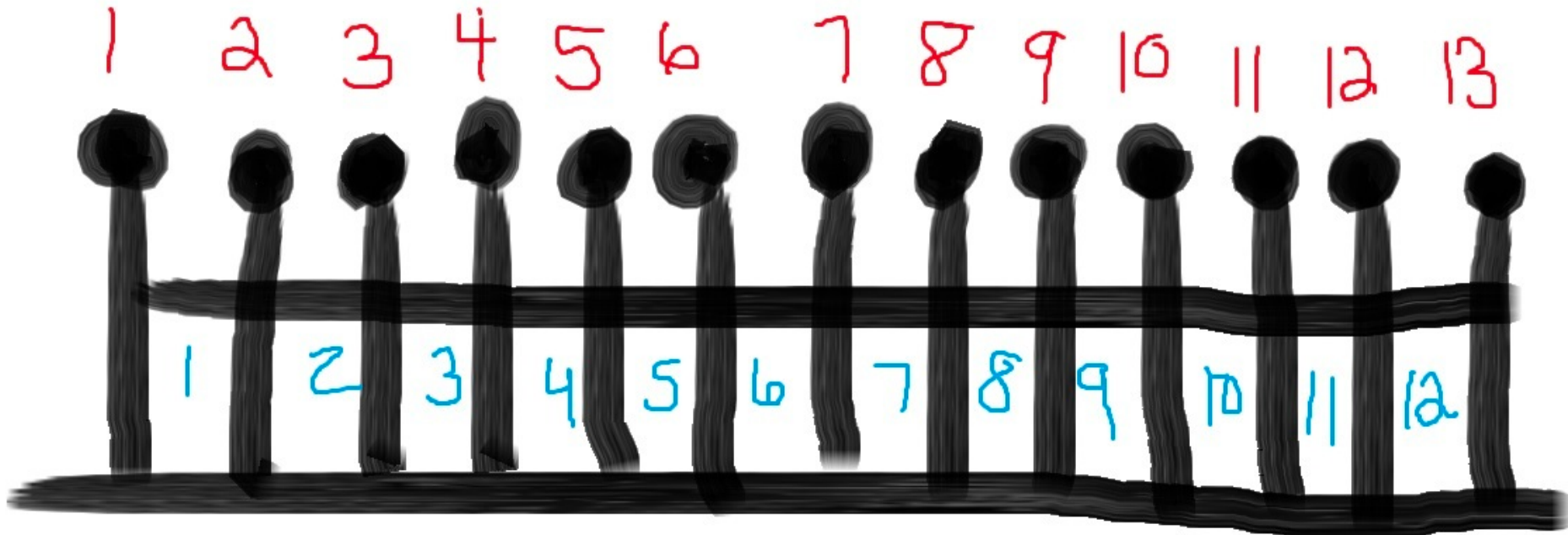
Place Beeper Line

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

Fixed!



Fence Post Problem

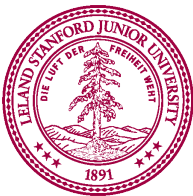


Also sometimes called an “Off By One Error”

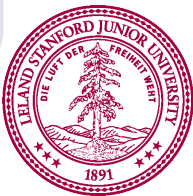
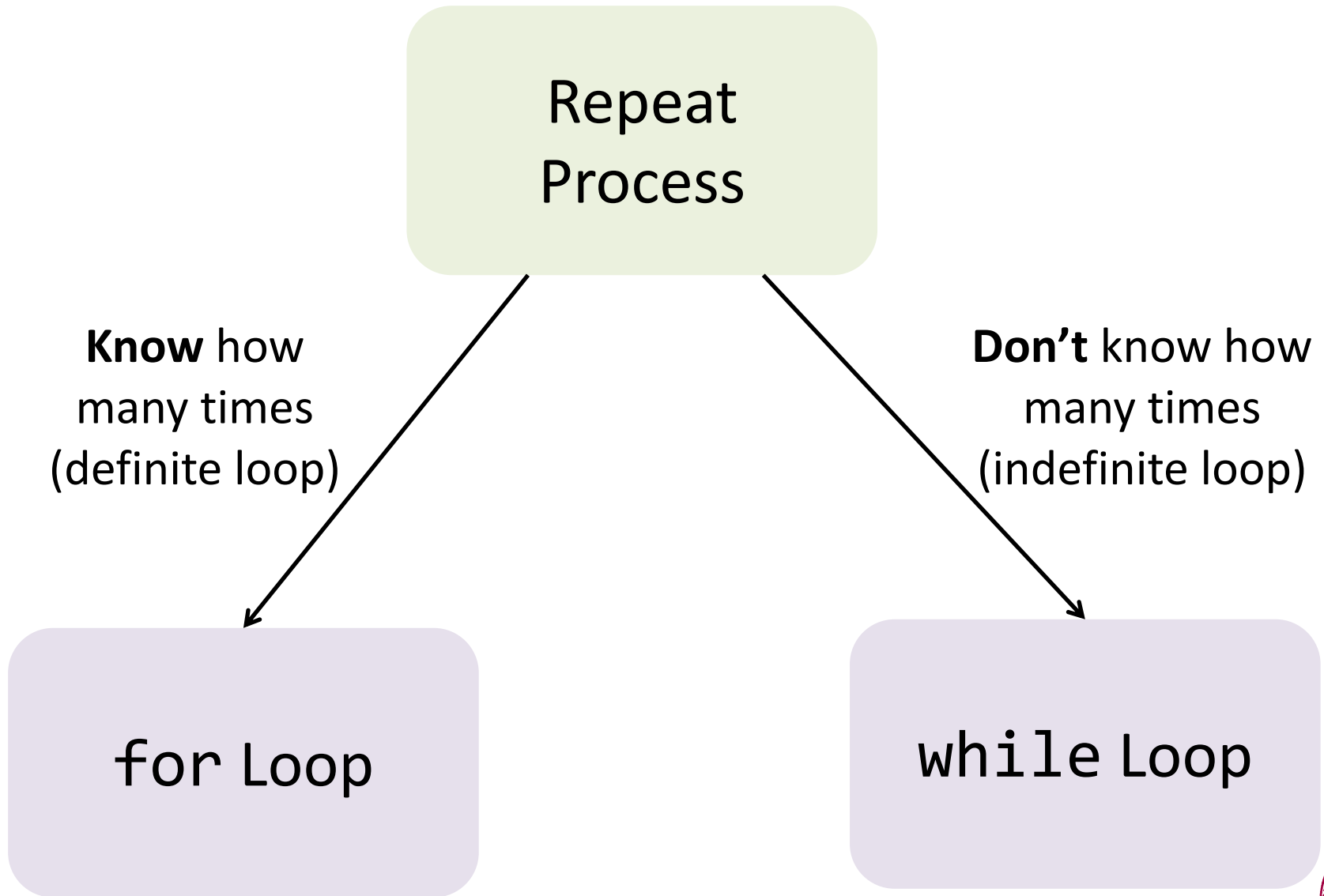


A program executes one line at a time.

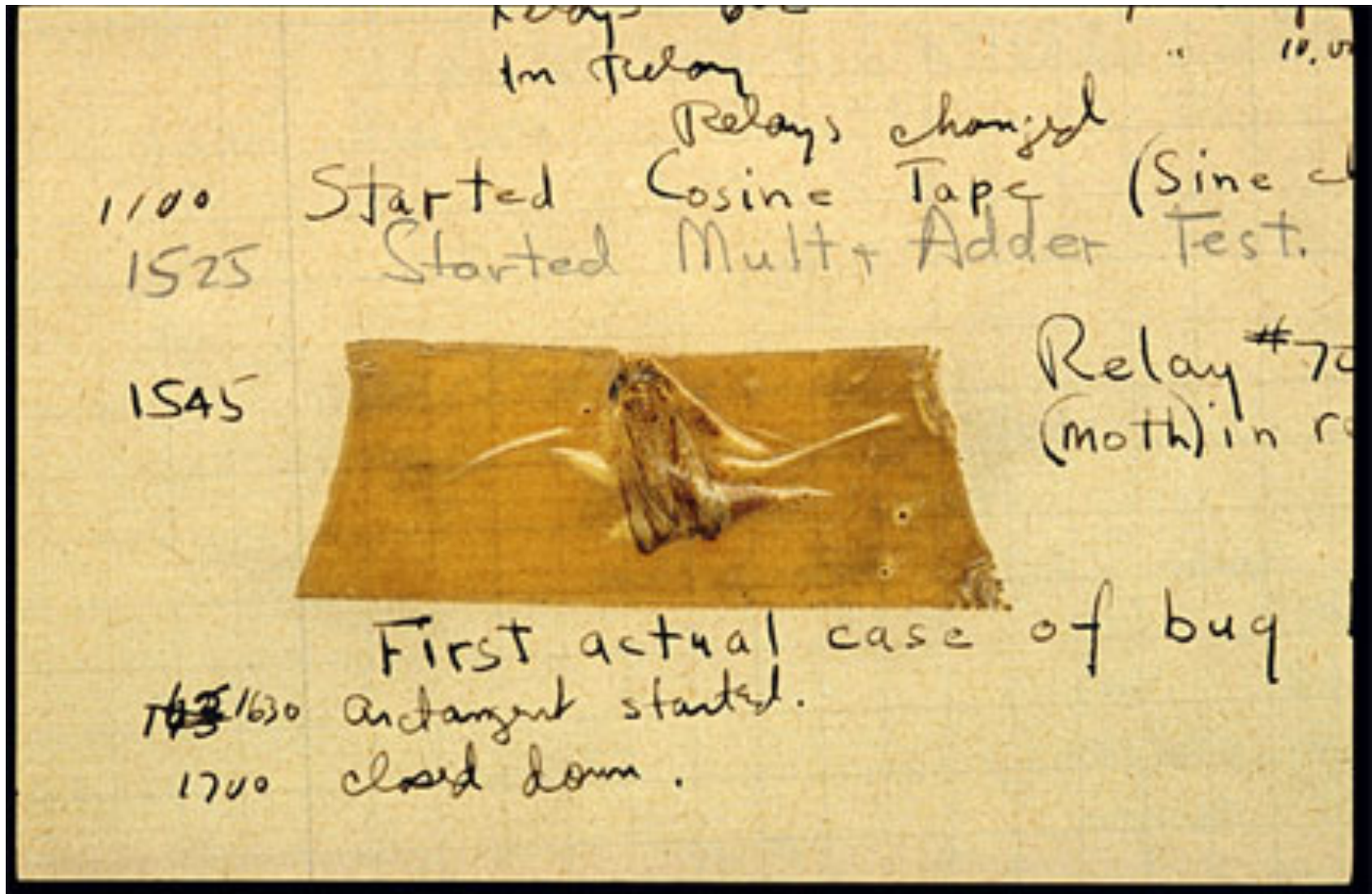
The **while** loop checks its condition only at the start of the code block and before repeating.



Which Loop



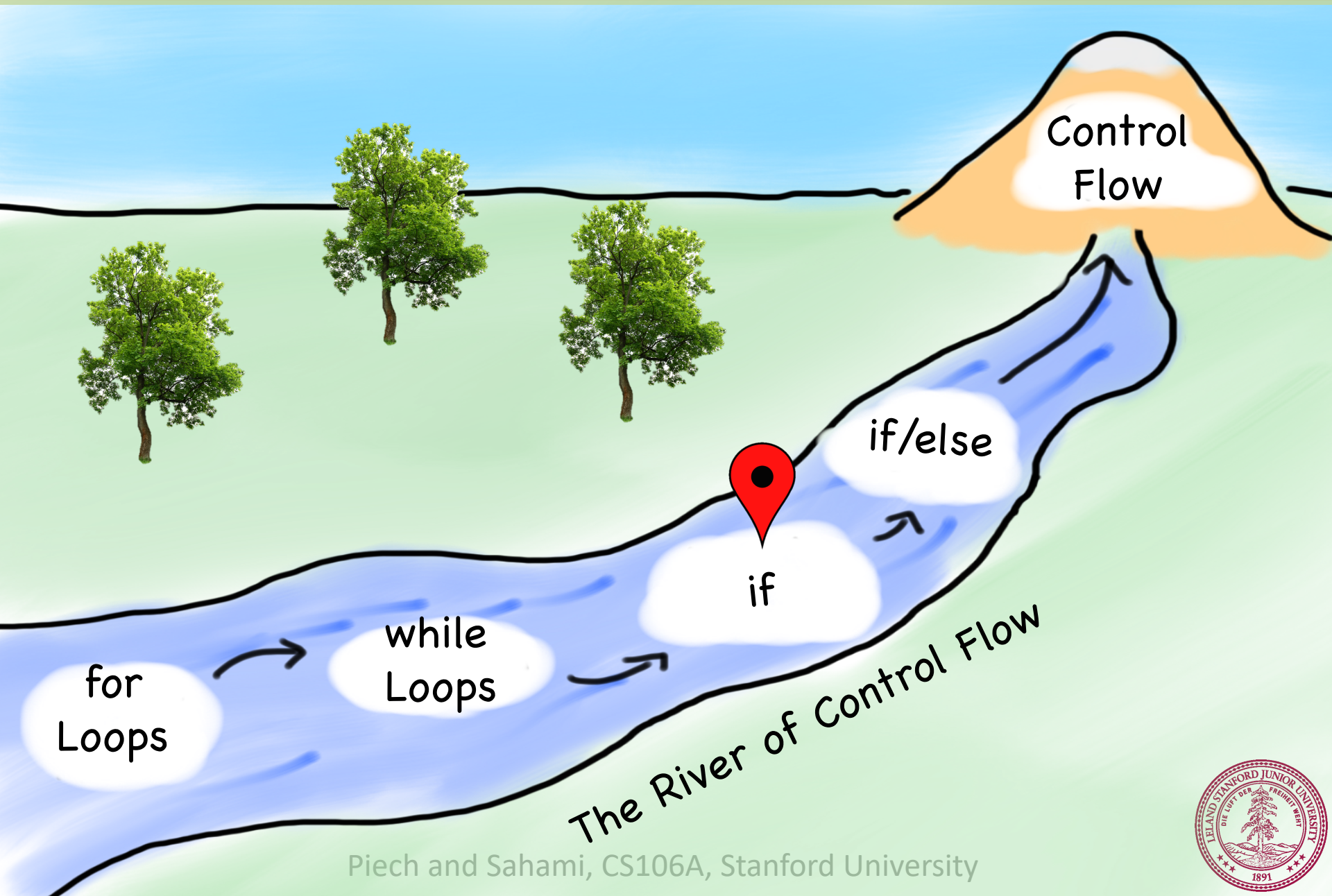
Actual Bug from Marc II



Grace Hopper

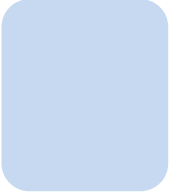


Today's Route



if statement

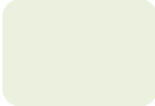
if *condition*:

 *statements*

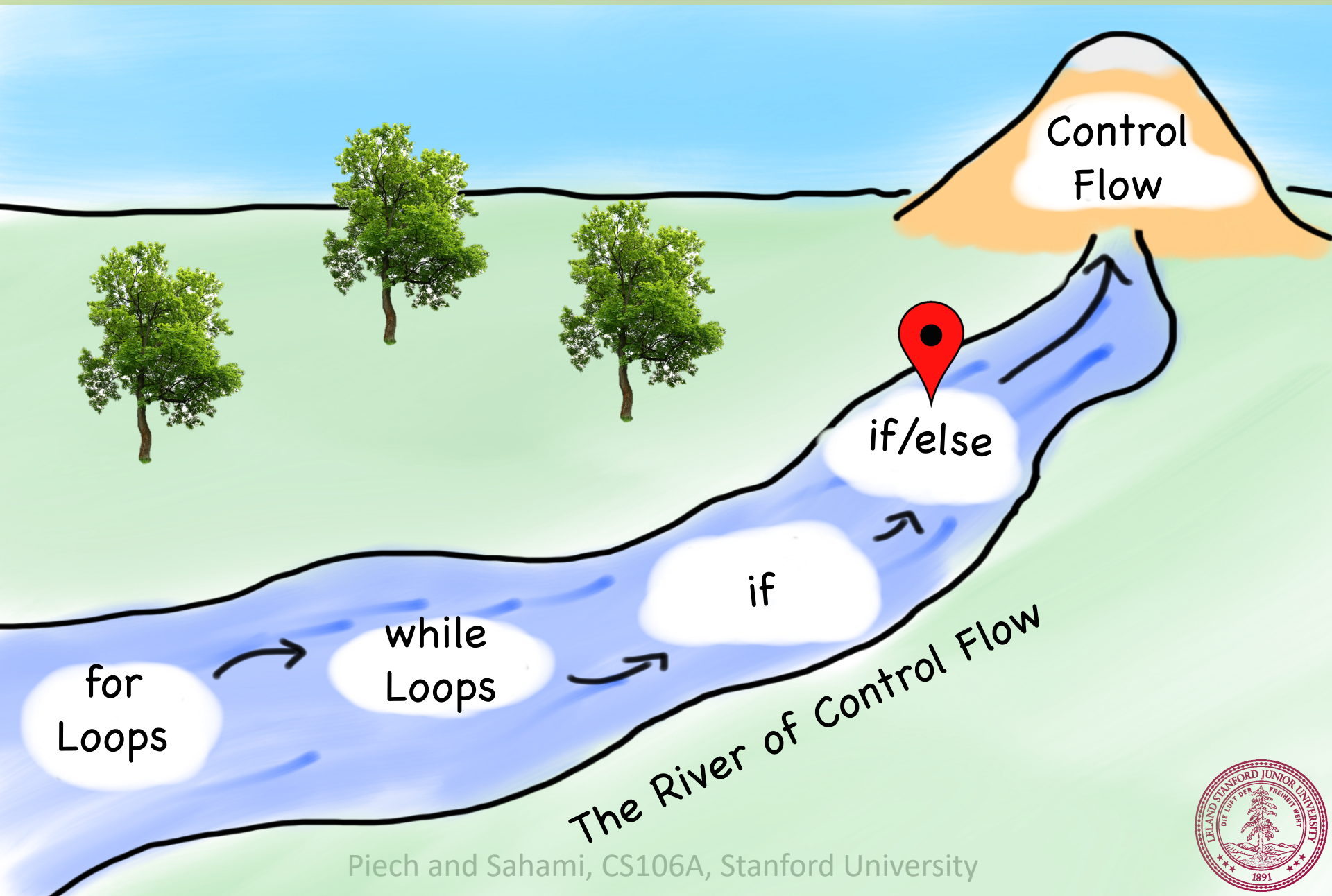
note indenting

def safe_pick_up():

 **if** beepers_present():

  pick_beeper() # note indenting

Today's Route

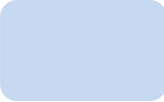


if-else statement

if *condition*:

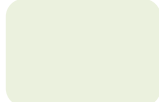
 *statements* # note indenting

else:

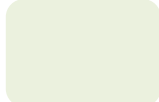
 *statements* # note indenting

def invert_beeper():

 **if** beepers_present():

 pick_beeper() # note indenting

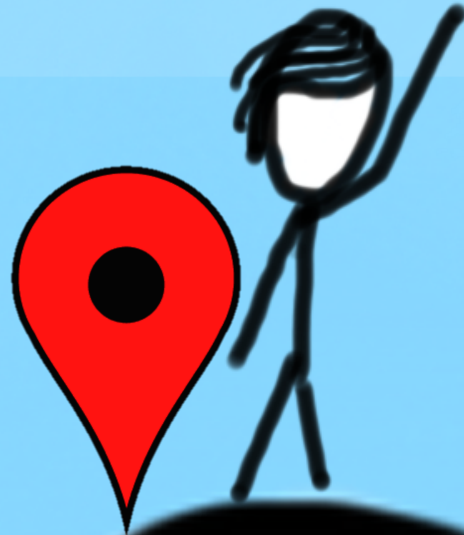
else:

 put_beeper() # note indenting

You just learned most of
programming "control flow"

Today's Goal

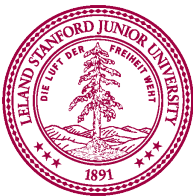
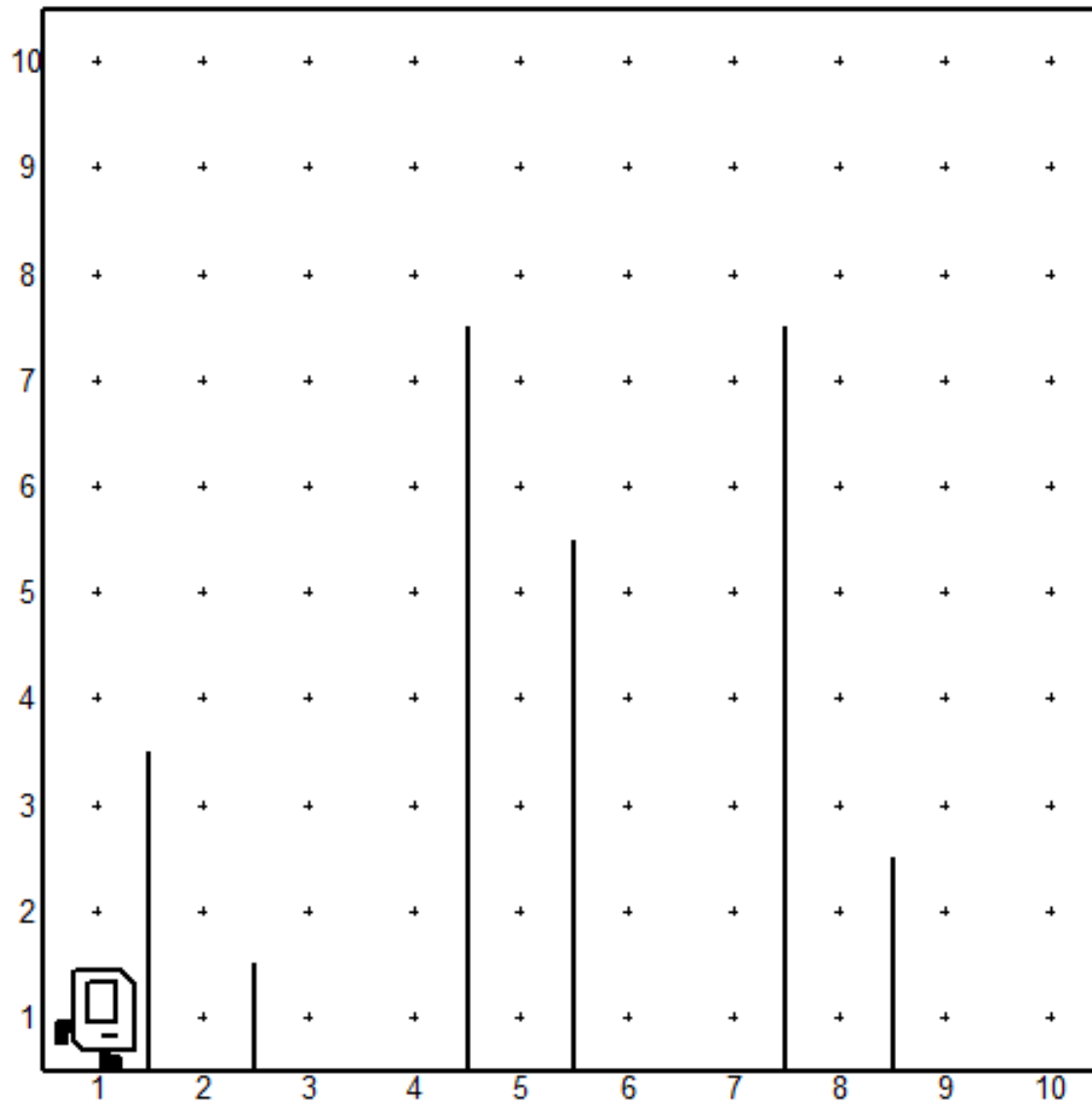
1. Code using loops and conditions
2. Trace programs that use loops and conditions



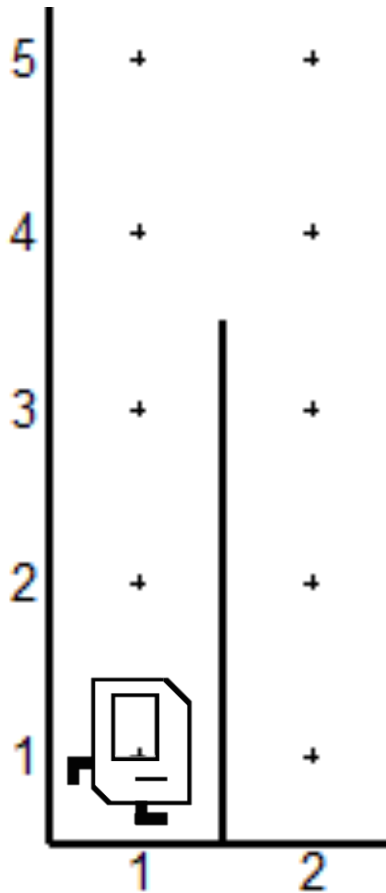


Putting it all together
SteepChaseKarel.py

Steeple Chase

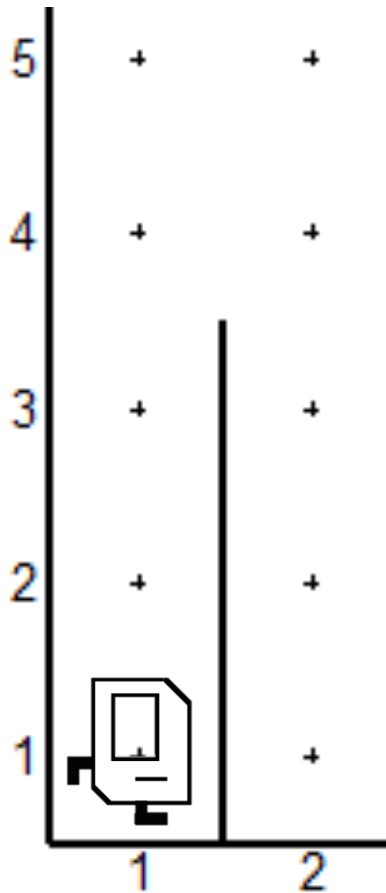


Focus on One Steeple



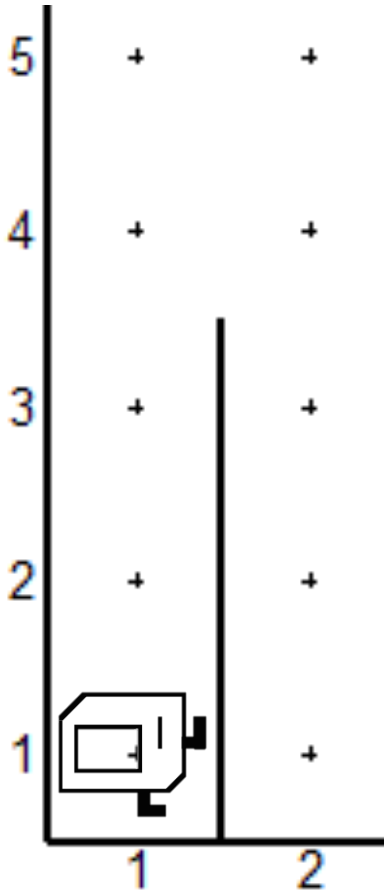
Focus on One Steeple

`turn_left()`



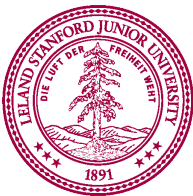
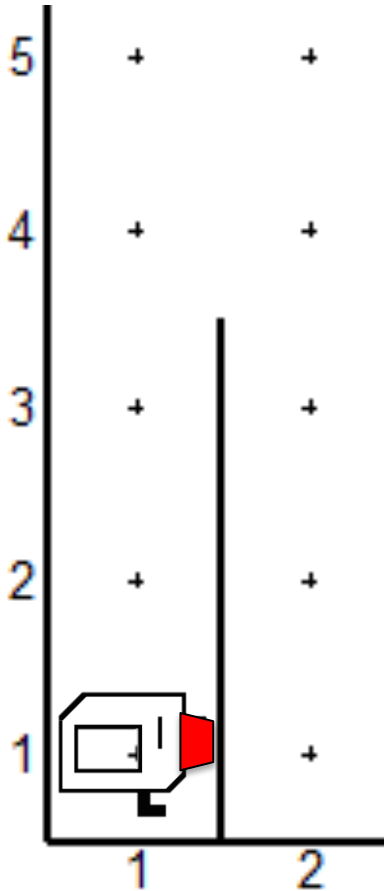
Focus on One Steeple

`turn_left()`



Focus on One Steeple

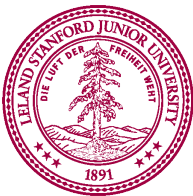
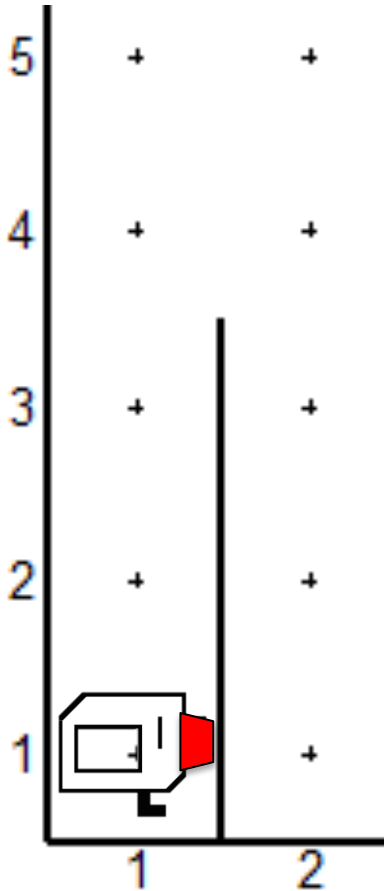
```
turn_left()  
while right_is_blocked():  
    move()
```



Focus on One Steeple

```
turn_left()
```

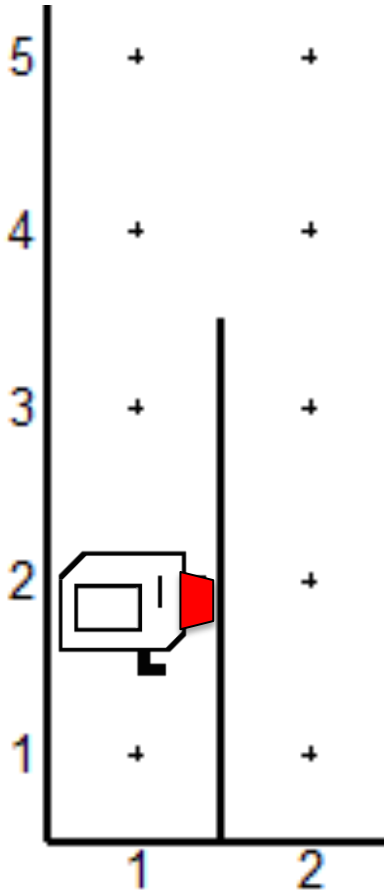
```
while right_is_blocked():  
    move()
```



Focus on One Steeple

```
turn_left()
```

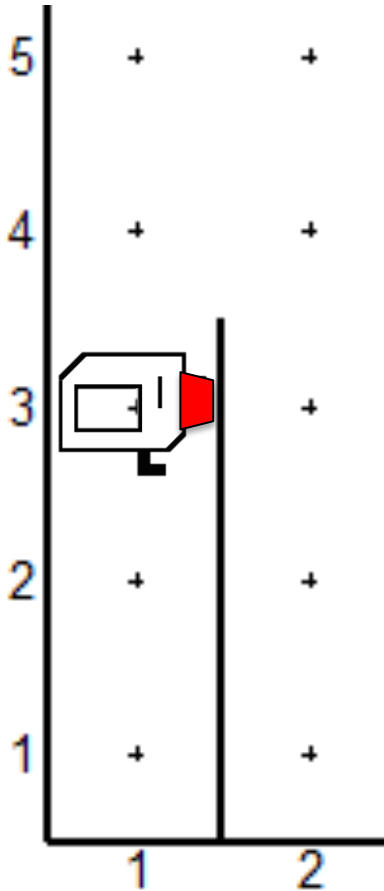
```
while right_is_blocked():  
    move()
```



Focus on One Steeple

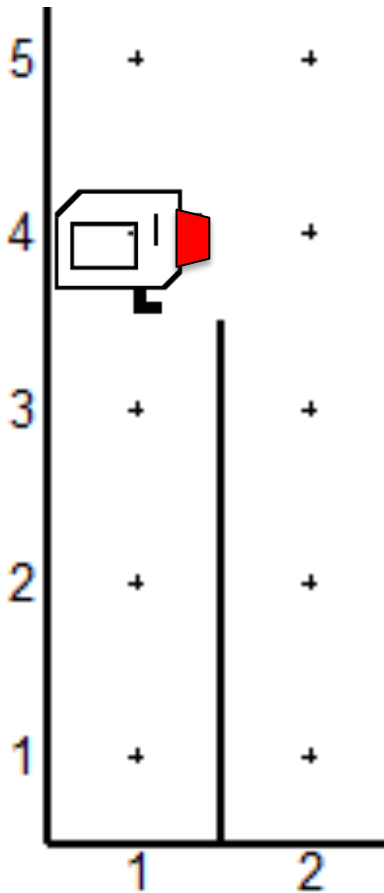
```
turn_left()
```

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



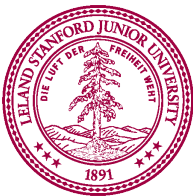
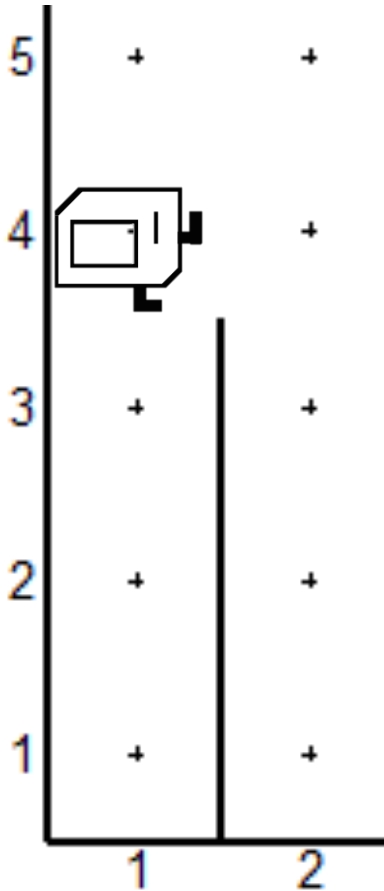
Focus on One Steeple

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



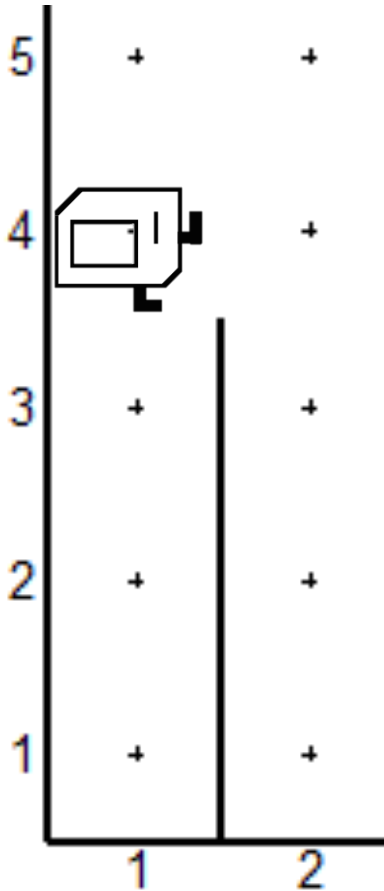
Focus on One Steeple

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



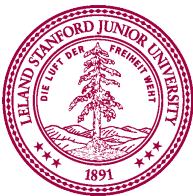
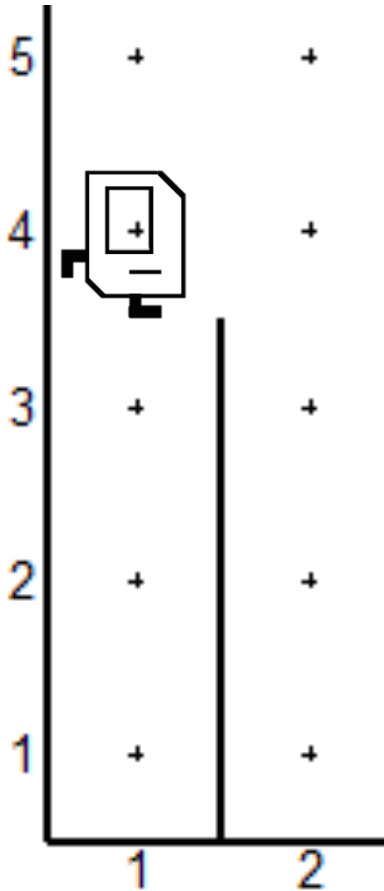
Focus on One Steeple

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



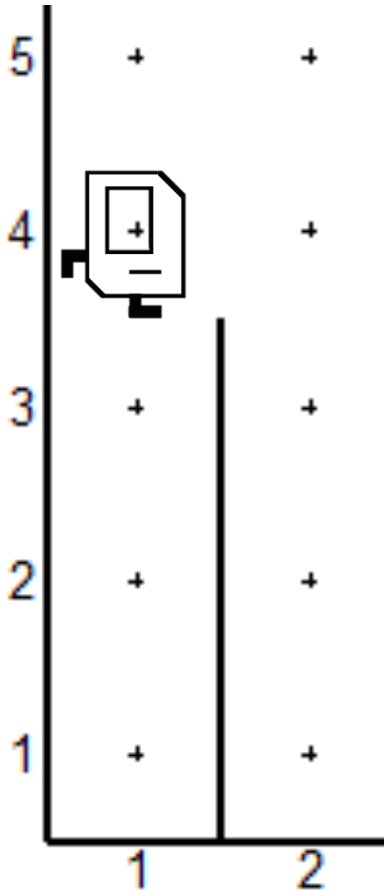
Focus on One Steeple

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



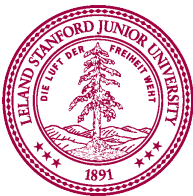
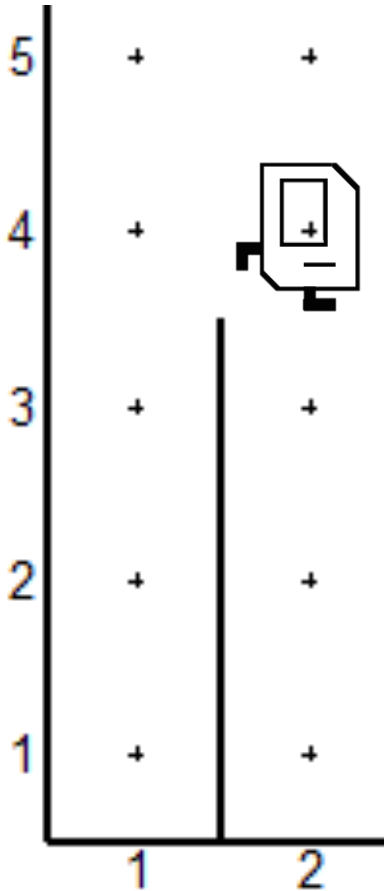
Focus on One Steeple

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

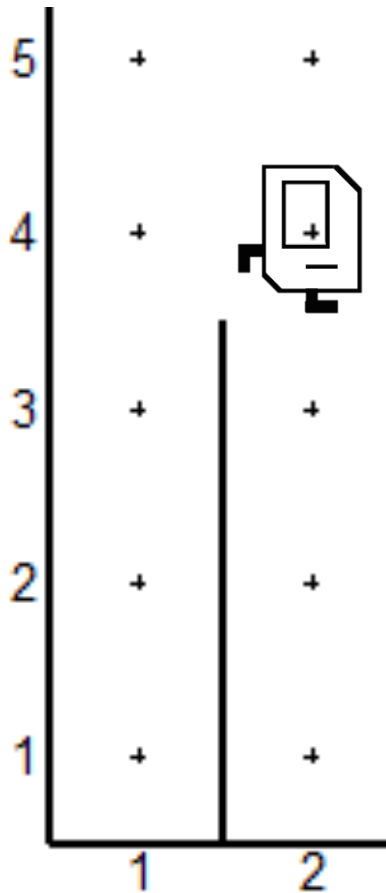


Focus on One Steeple

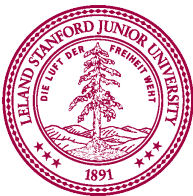
```
turn_left()  
while right_is_blocked():  
    move()  
turn_right()  
move()
```



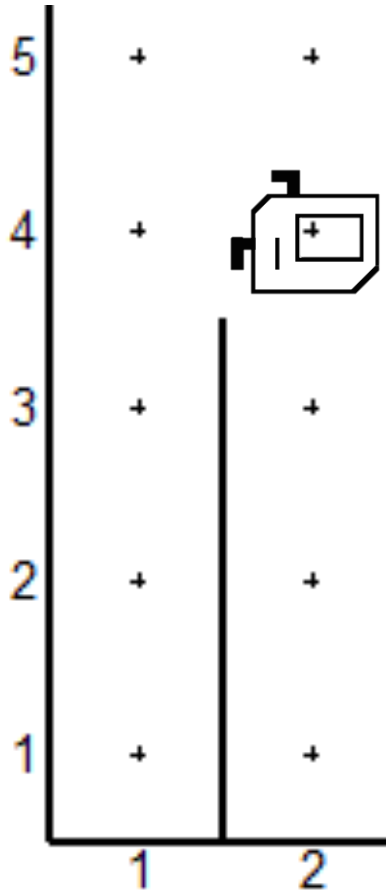
Focus on One Steeple



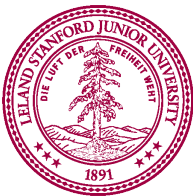
```
turn_left()  
while right_is_blocked():  
    move()  
turn_right()  
move()  
turn_right()
```



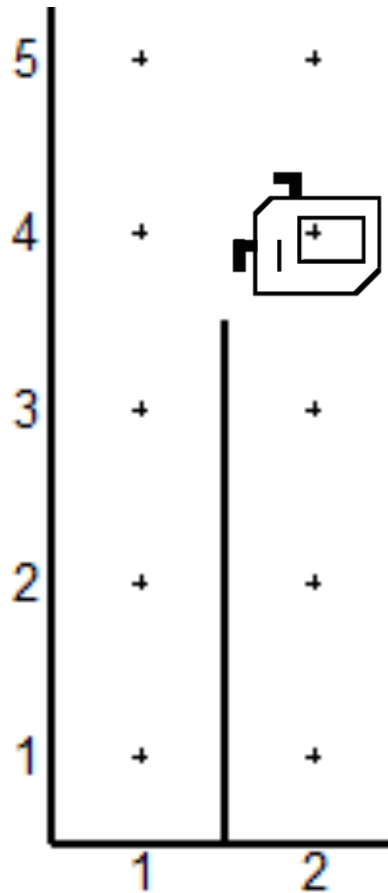
Focus on One Steeple



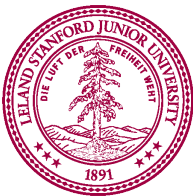
```
turn_left()  
while right_is_blocked():  
    move()  
turn_right()  
move()  
turn_right()
```



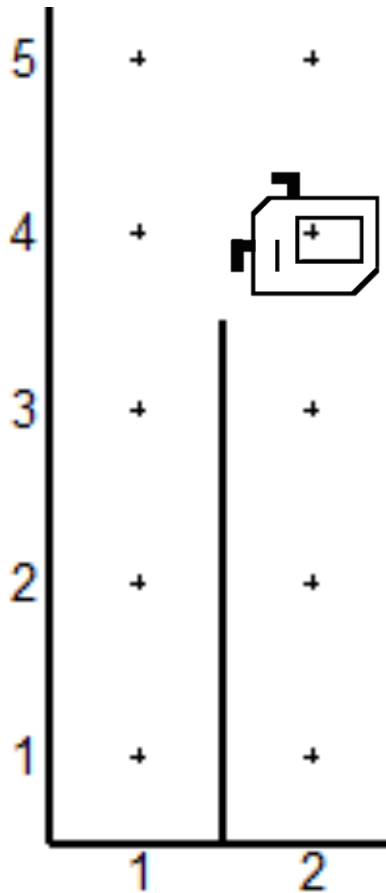
Focus on One Steeple



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

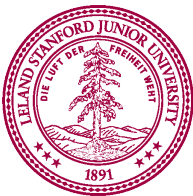


Focus on One Steeple

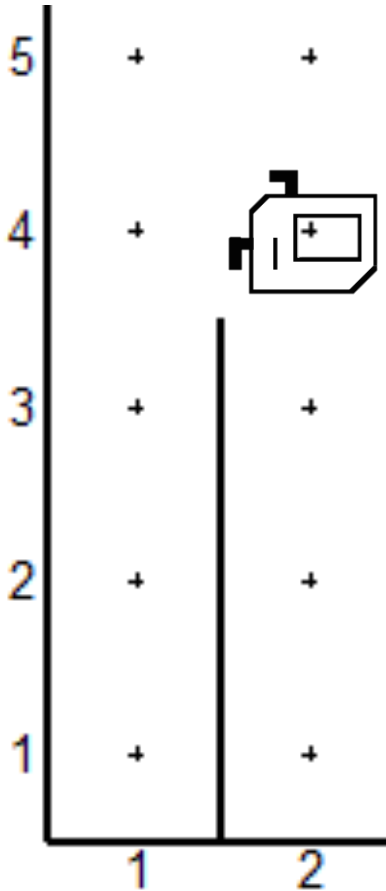


```
turn_left()  
while right_is_blocked():  
    move()  
turn_right()  
move()  
turn_right()  
move_to_wall()
```

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

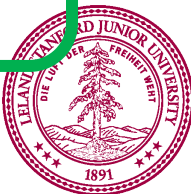


Focus on One Steeple

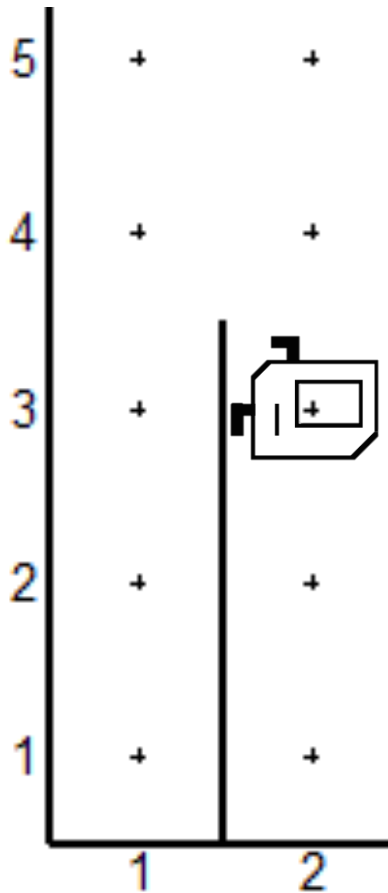


```
turn_left()
while right_is_blocked():
    move()
turn_right()
move()
turn_right()
move to wall()
```

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

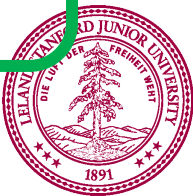


Focus on One Steeple

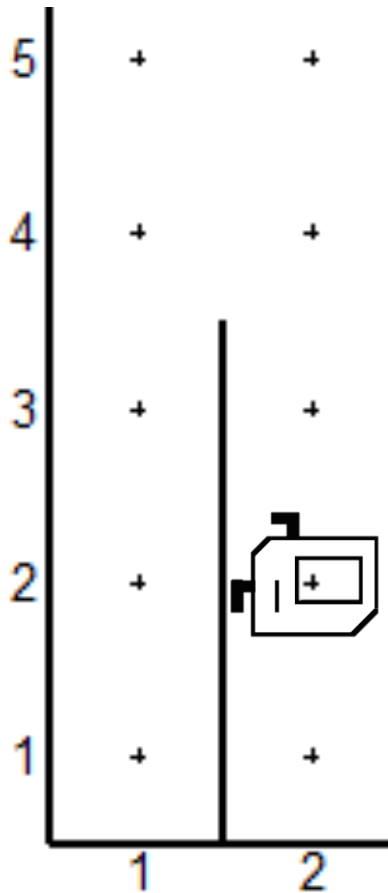


```
turn_left()
while right_is_blocked():
    move()
turn_right()
move()
turn_right()
move_to_wall()
```

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

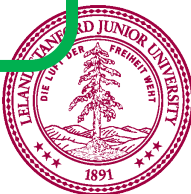


Focus on One Steeple

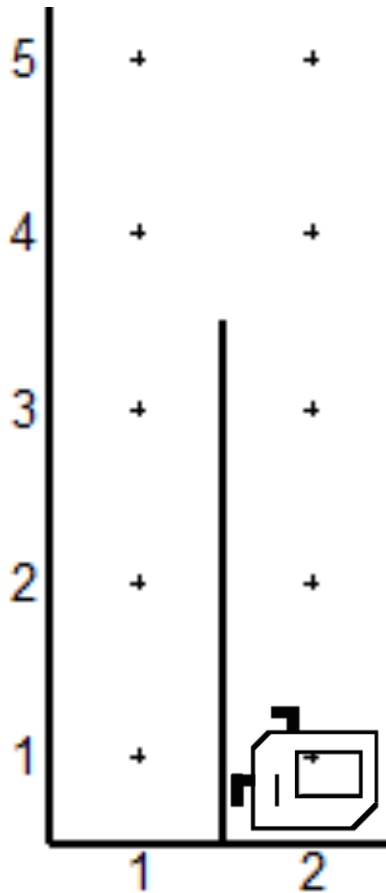


```
turn_left()  
while right_is_blocked():  
    move()  
turn_right()  
move()  
turn_right()  
move_to_wall()
```

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

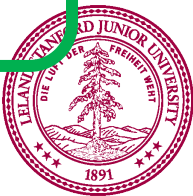


Focus on One Steeple

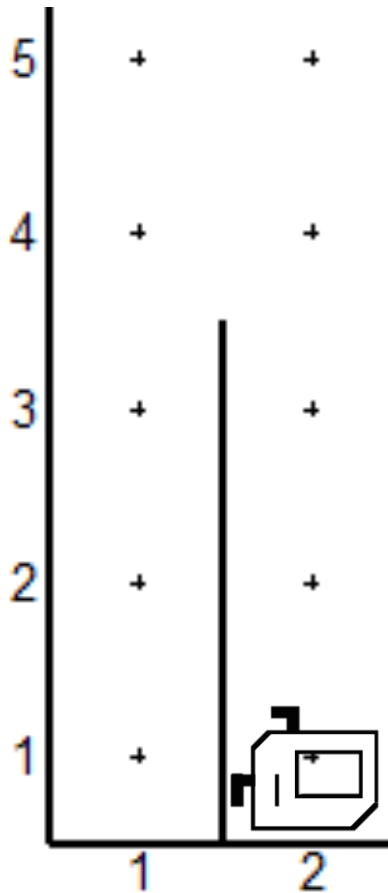


```
turn_left()  
while right_is_blocked():  
    move()  
turn_right()  
move()  
turn_right()  
move_to_wall()
```

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

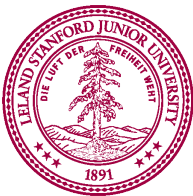


Focus on One Steeple

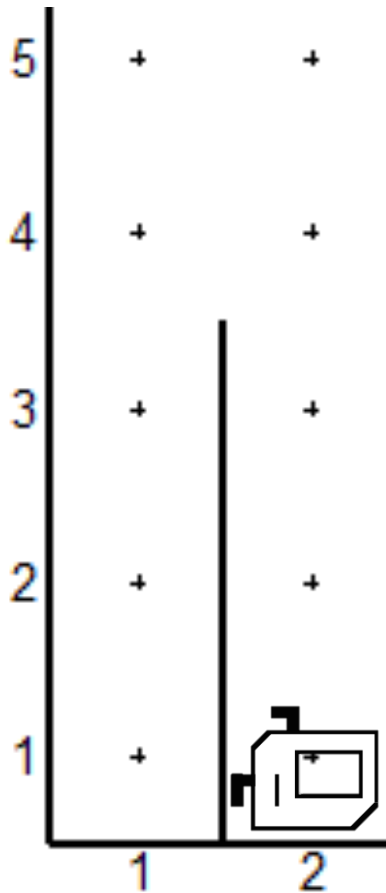


```
turn_left()
while right_is_blocked():
    move()
turn_right()
move()
turn_right()
move_to_wall()
```

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

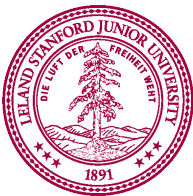


Focus on One Steeple

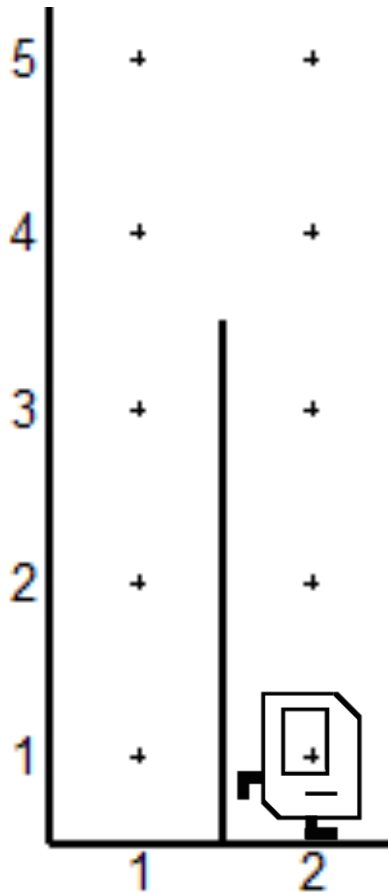


```
turn_left()
while right_is_blocked():
    move()
turn_right()
move()
turn_right()
move_to_wall()
turn_left()
```

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

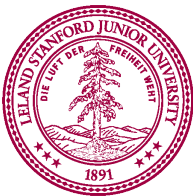


Focus on One Steeple

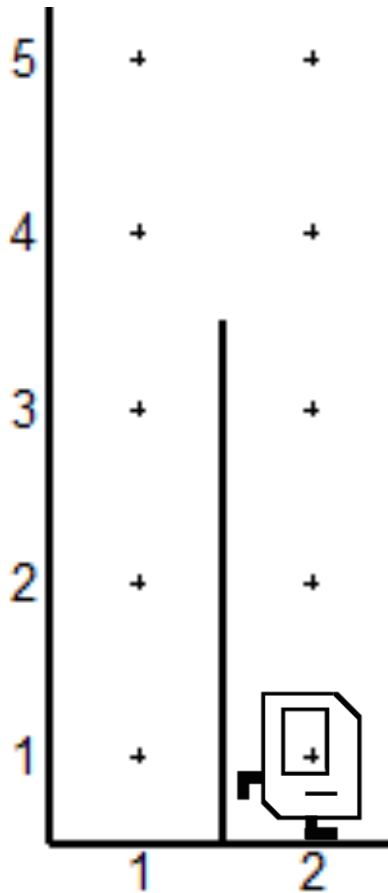


```
turn_left()  
while right_is_blocked():  
    move()  
turn_right()  
move()  
turn_right()  
move_to_wall()  
turn_left()
```

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



Focus on One Steeple



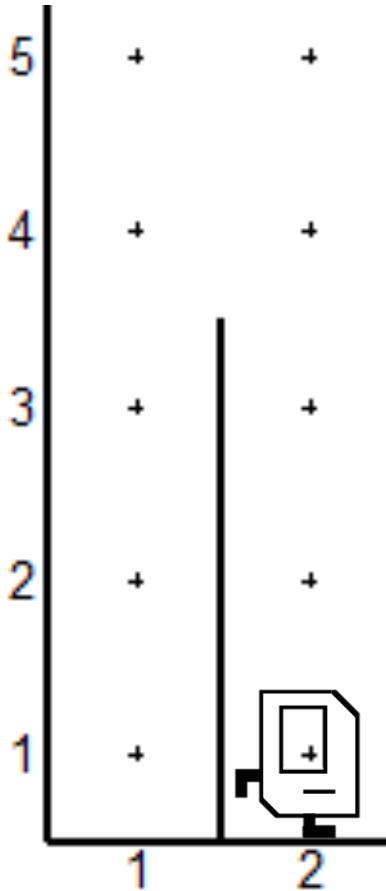
```
turn_left()
while right_is_blocked():
    move()
turn_right()
move()
turn_right()
move_to_wall()
turn_left()
```

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()
```

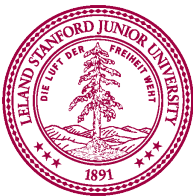
```
descend_hurdle()
```



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()`

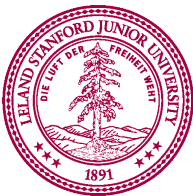


Focus on One Steeple

```
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