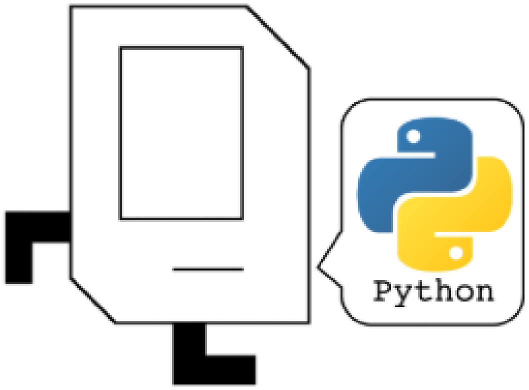
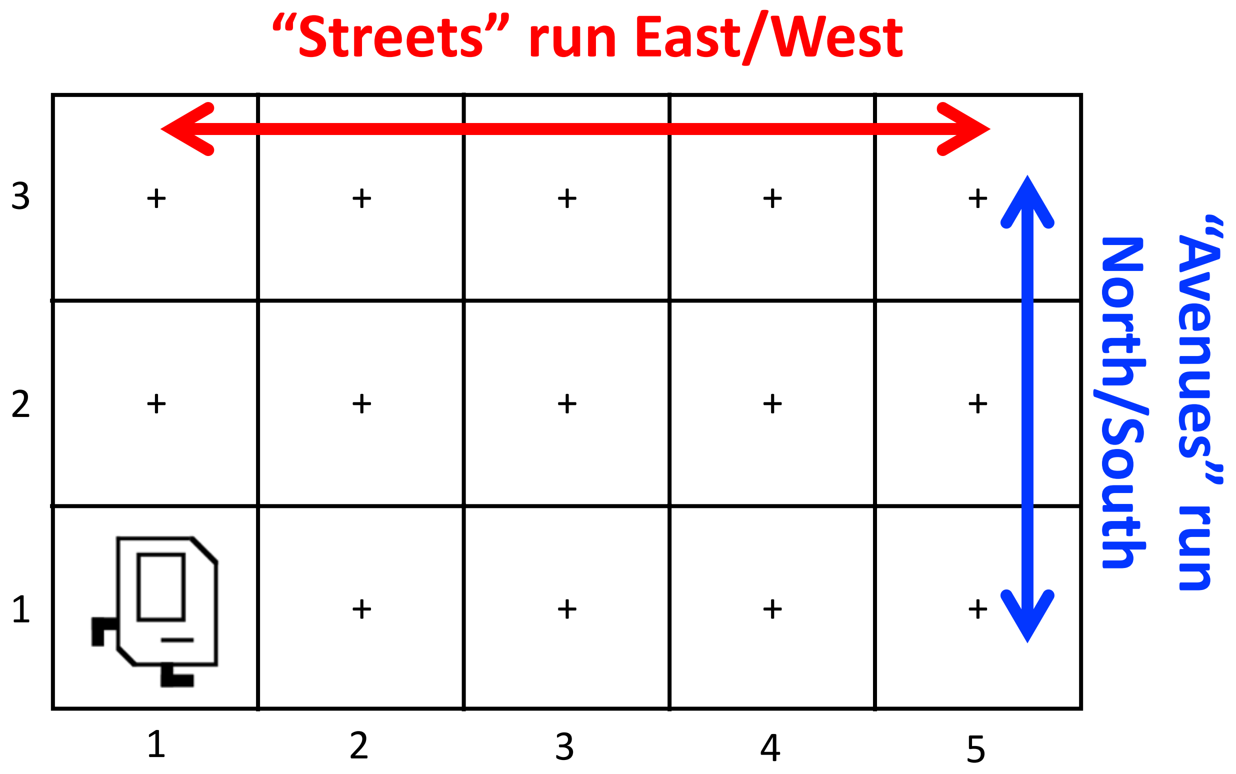
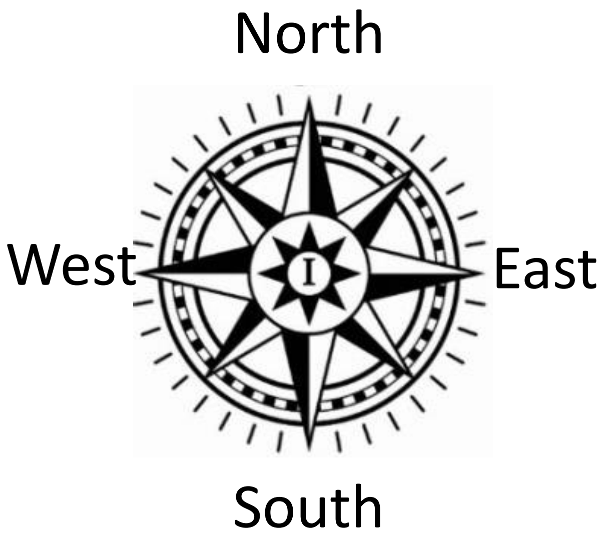


- Reading: Should read the “Karel Reader”
- “Using Karel with VS Code”
  - Tells you how to get started with writing Karel programs
- “Assignment 1”
  - Set of Karel programs for you to write
- Only use features of Karel in the course reader
  - No other features of Python may be used in Karel programs!

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***

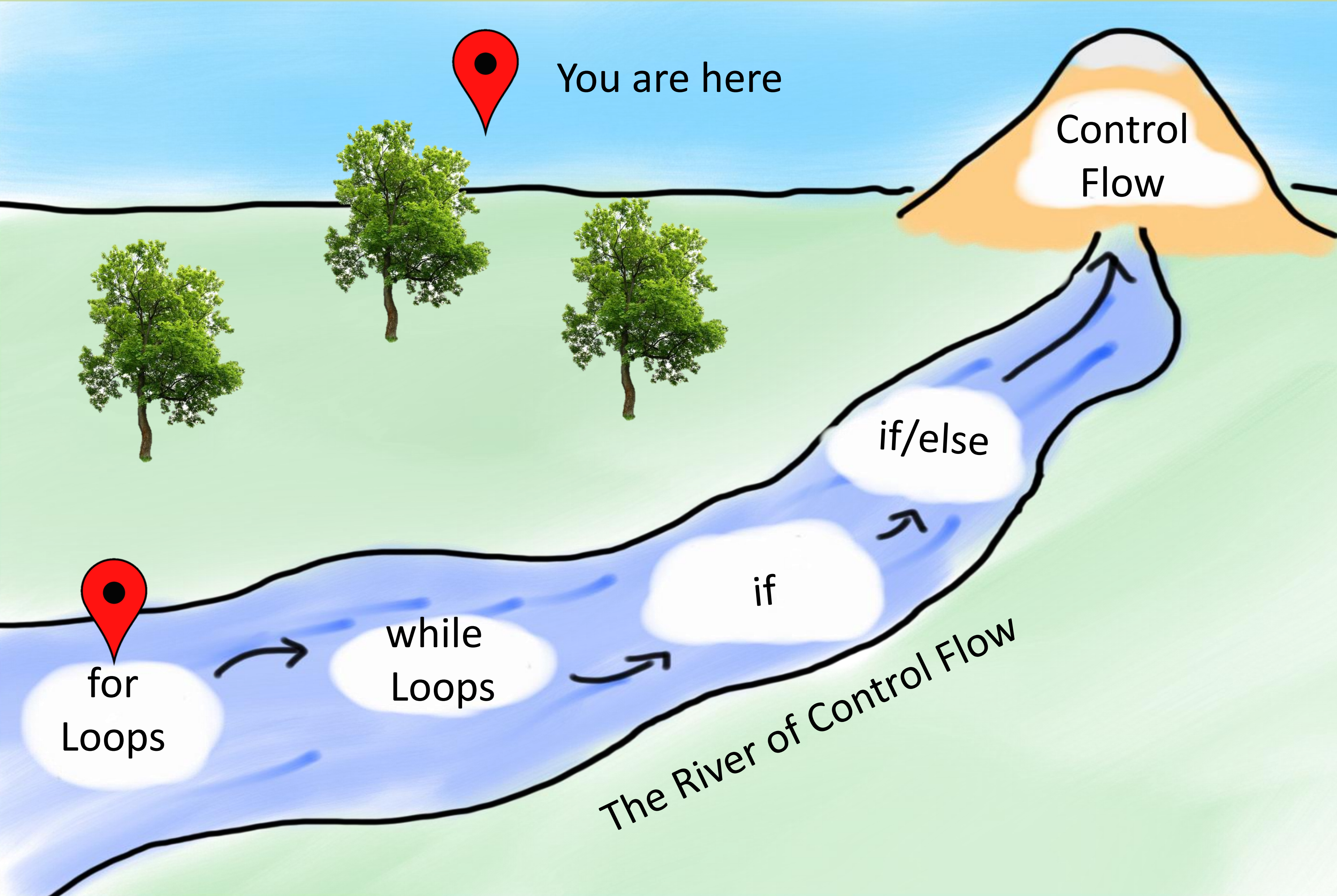
One line *comment*

Descriptive *names* (snake_case)

Today's Goal

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

Today's Route



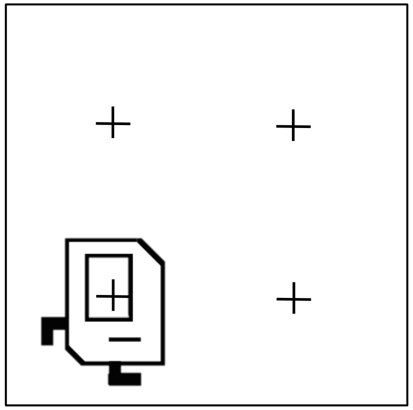
for loop

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

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

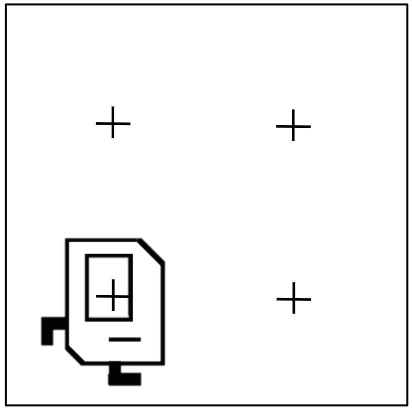
Place Beeper Square

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



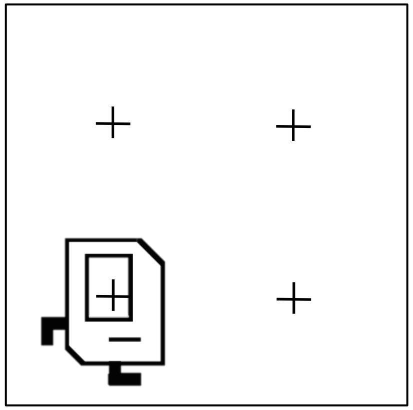
Place Beeper Square

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



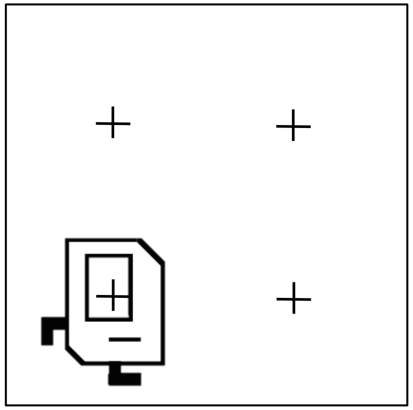
Place Beeper Square

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



Place Beeper Square

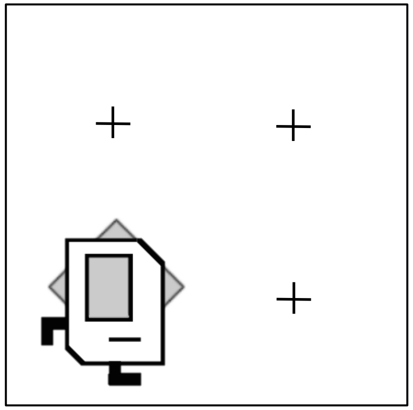
```
def main():
    for i in range(4):
        put_beeper()
        move()
        turn_left()
```



First time through the loop

Place Beeper Square

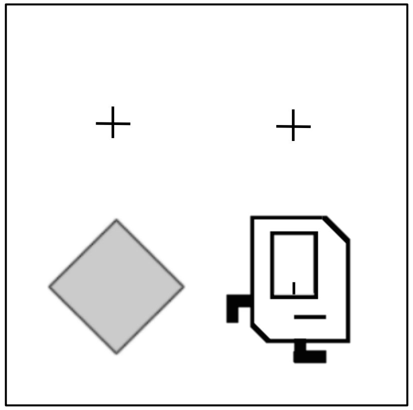
```
def main():
    for i in range(4):
        put_beeper()
        move()
        turn_left()
```



First time through the loop

Place Beeper Square

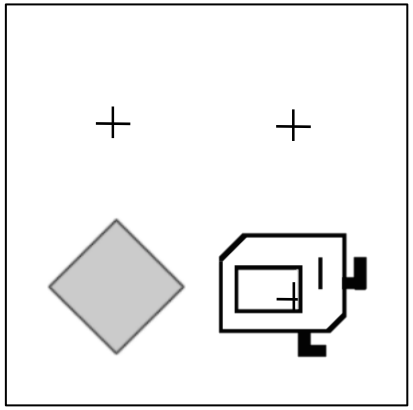
```
def main():
    for i in range(4):
        put_beeper()
        move()
        turn_left()
```



First time through the loop

Place Beeper Square

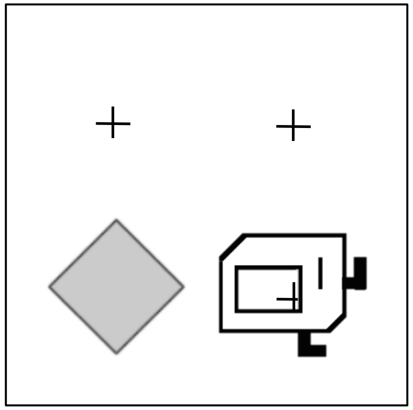
```
def main():
    for i in range(4):
        put_beeper()
        move()
        turn_left()
```



First time through the loop

Place Beeper Square

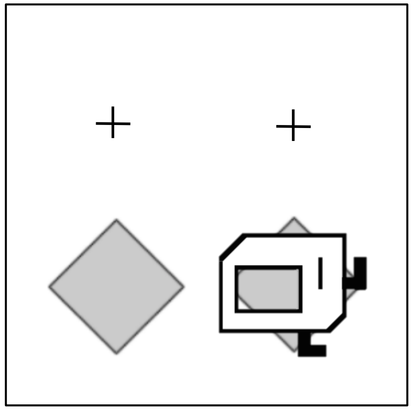
```
def main():
    for i in range(4):
        put_beeper()
        move()
        turn_left()
```



Second time through the loop

Place Beeper Square

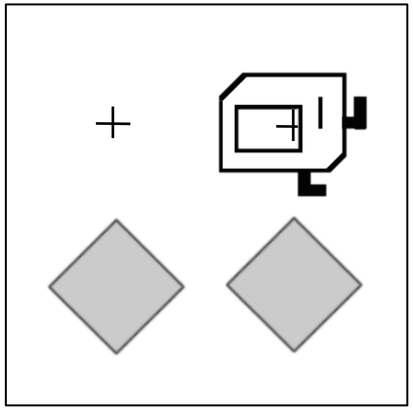
```
def main():
    for i in range(4):
        put_beeper()
        move()
        turn_left()
```



Second time through the loop

Place Beeper Square

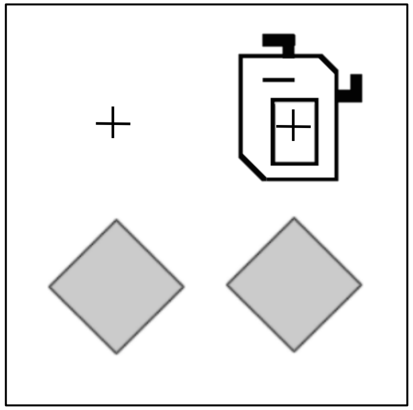
```
def main():
    for i in range(4):
        put_beeper()
        move()
        turn_left()
```



Second time through the loop

Place Beeper Square

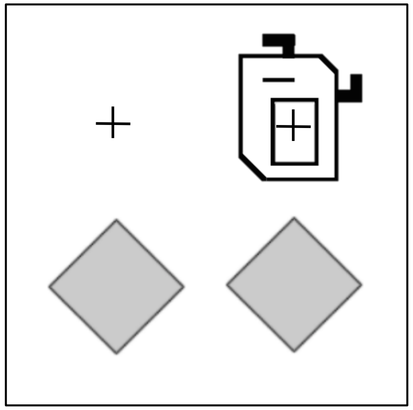
```
def main():
    for i in range(4):
        put_beeper()
        move()
        turn_left()
```



Second time through the loop

Place Beeper Square

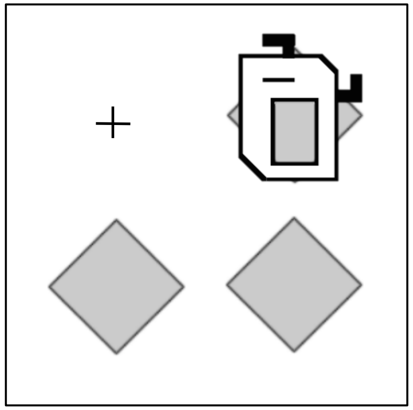
```
def main():
    for i in range(4):
        put_beeper()
        move()
        turn_left()
```



Third time through the loop

Place Beeper Square

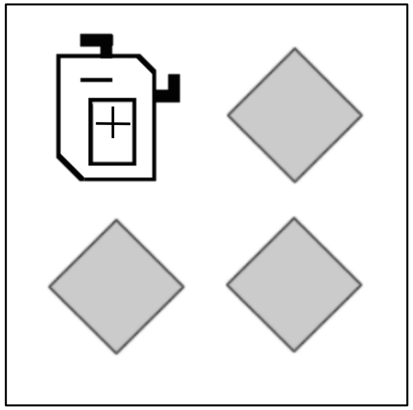
```
def main():
    for i in range(4):
        put_beeper()
        move()
        turn_left()
```



Third time through the loop

Place Beeper Square

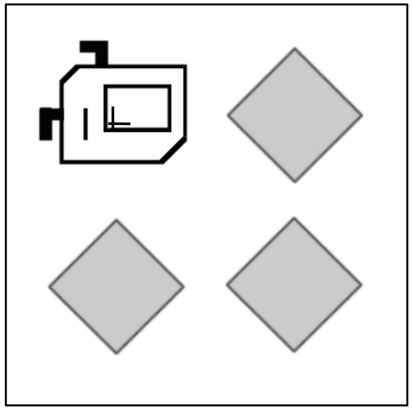
```
def main():
    for i in range(4):
        put_beeper()
        move()
        turn_left()
```



Third time through the loop

Place Beeper Square

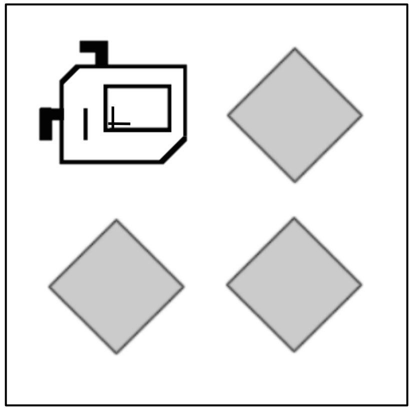
```
def main():
    for i in range(4):
        put_beeper()
        move()
        turn_left()
```



Third time through the loop

Place Beeper Square

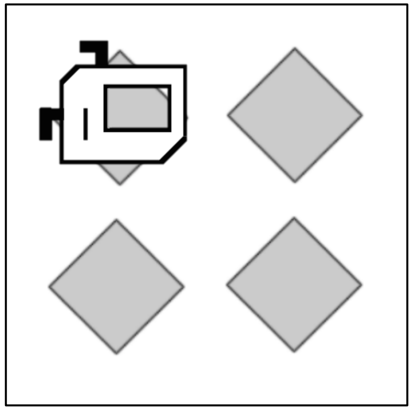
```
def main():
    for i in range(4):
        put_beeper()
        move()
        turn_left()
```



Fourth time through the loop

Place Beeper Square

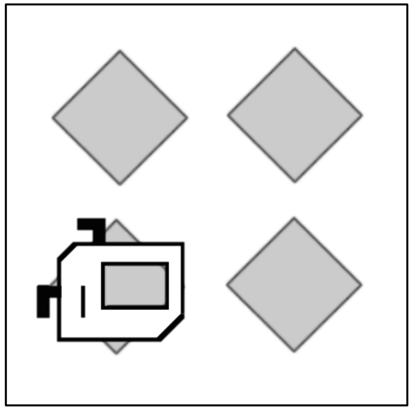
```
def main():
    for i in range(4):
        put_beeper()
        move()
        turn_left()
```



Fourth time through the loop

Place Beeper Square

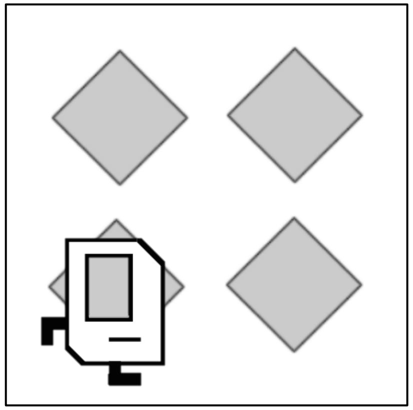
```
def main():
    for i in range(4):
        put_beeper()
        move()
        turn_left()
```



Fourth time through the loop

Place Beeper Square

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



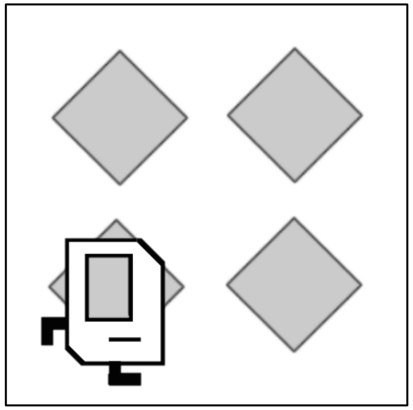
Fourth time through the loop

Place Beeper Square

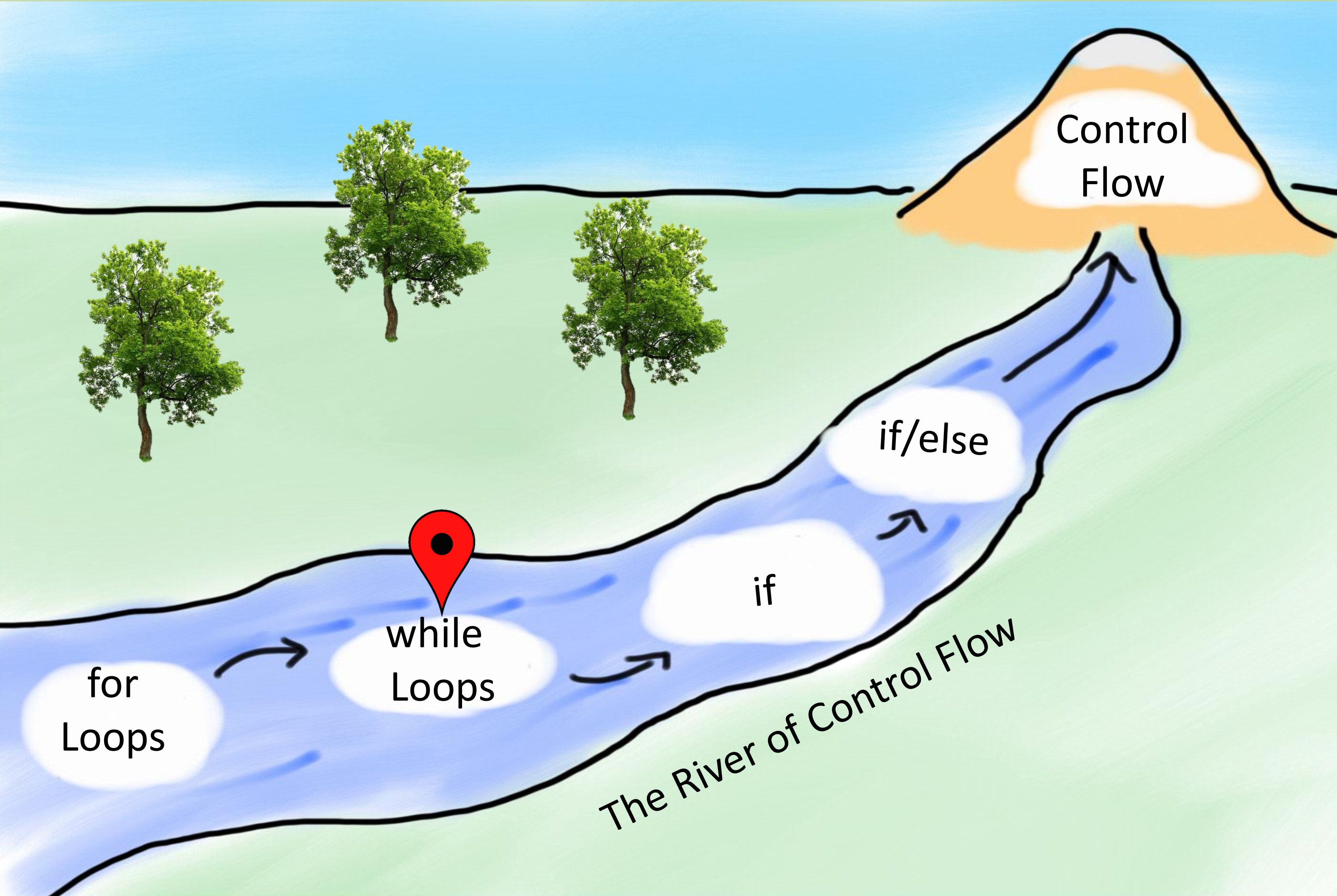
```
def main():
    for i in range(4):
        put_beeper()
        move()
        turn_left()
```

You often want 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
```

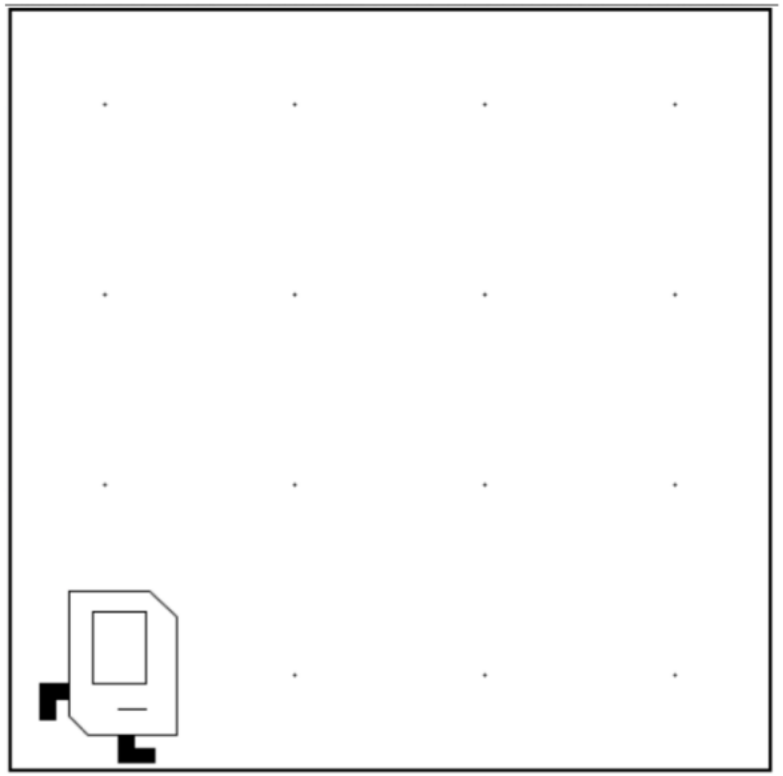
Conditions Karel Can Check For

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

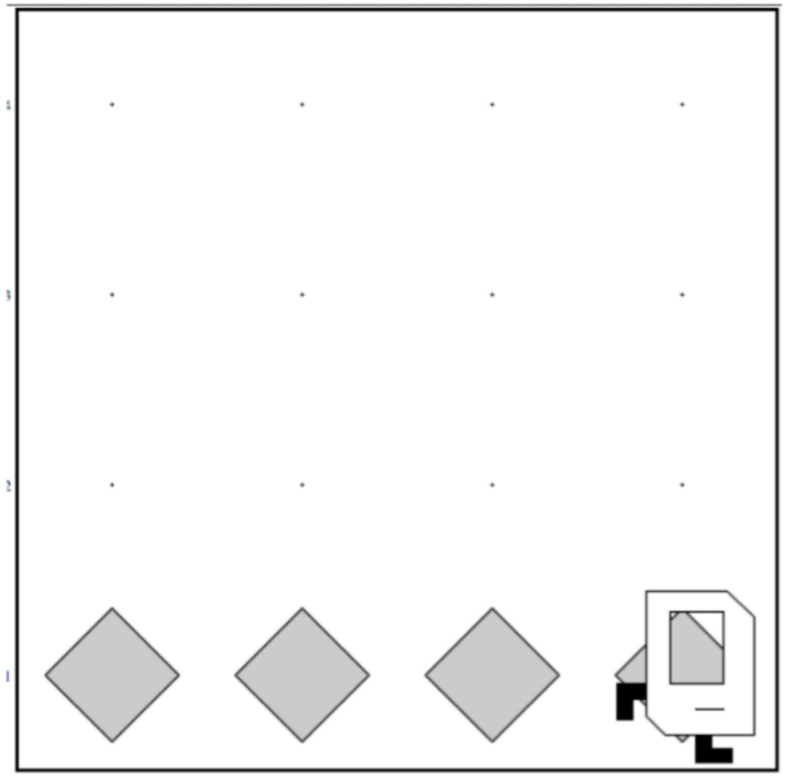
This is in Chapter 10 of the online Karel course reader

Task: Place Beeper Line

Before



After

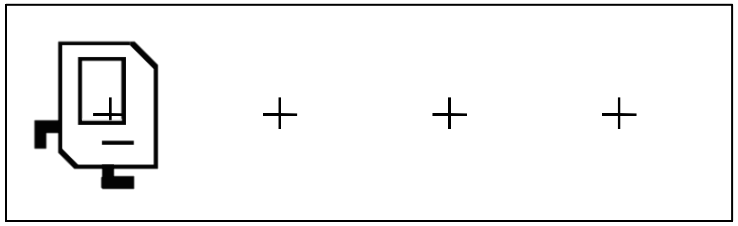


Place Beeper Line

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

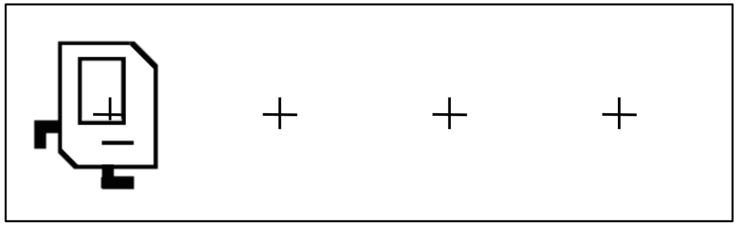
Place Beeper Line

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



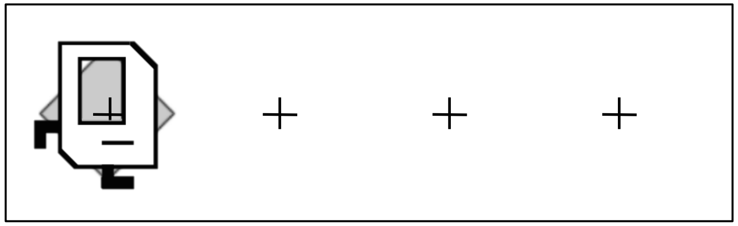
Place Beeper Line

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



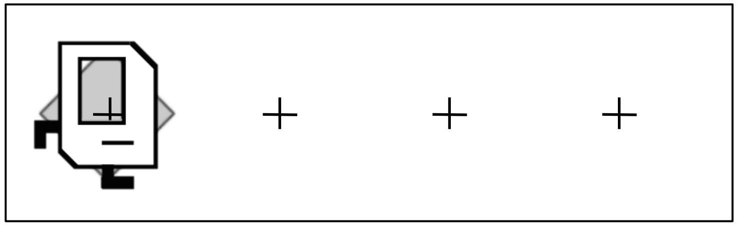
Place Beeper Line

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



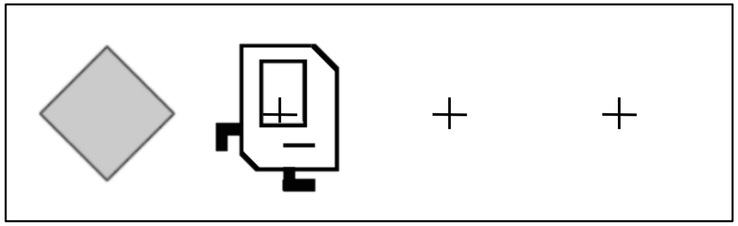
Place Beeper Line

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



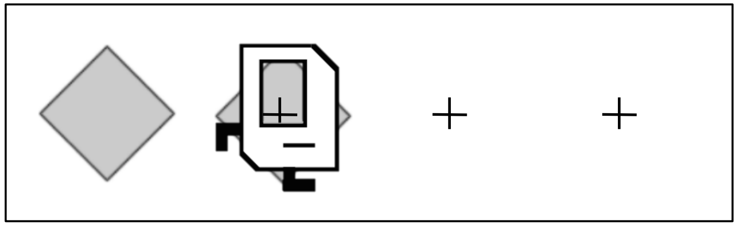
Place Beeper Line

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



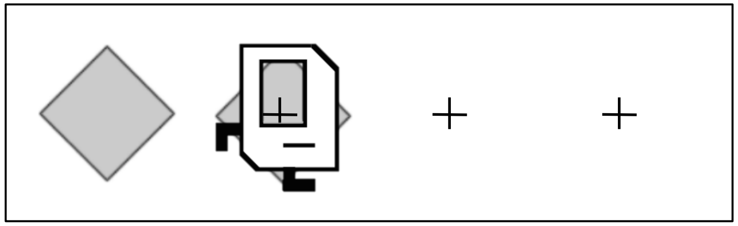
Place Beeper Line

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



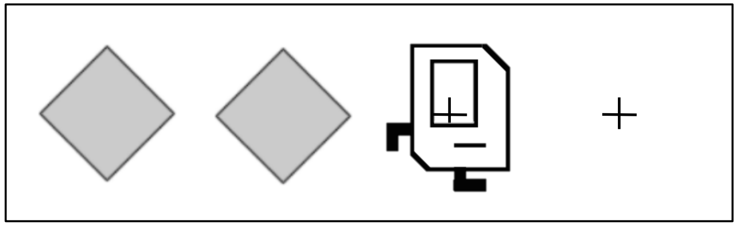
Place Beeper Line

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



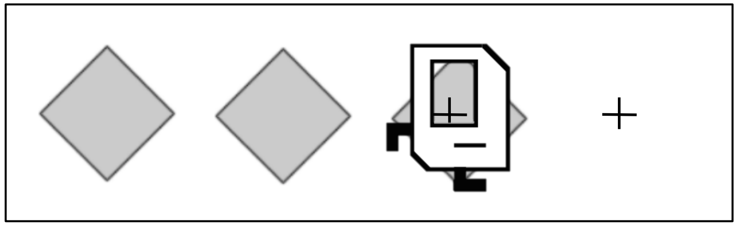
Place Beeper Line

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



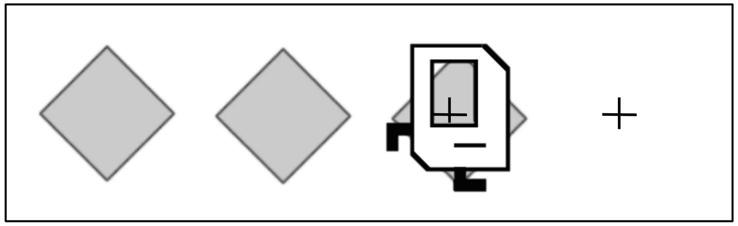
Place Beeper Line

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



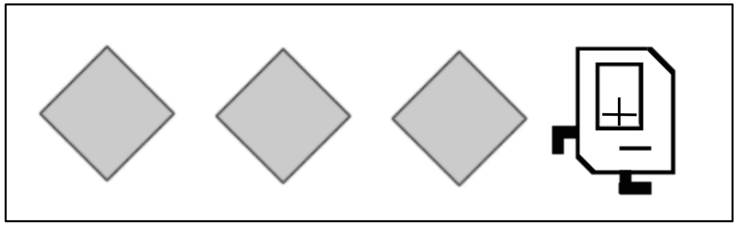
Place Beeper Line

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



Place Beeper Line

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

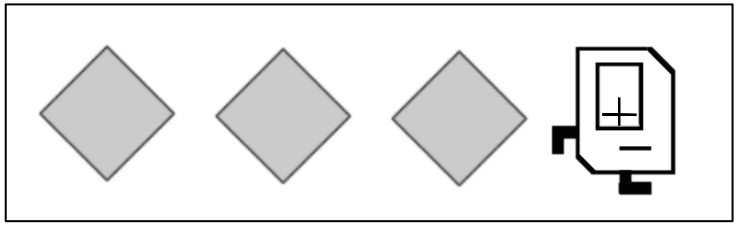


Place Beeper Line

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

BUGGY!

Done!

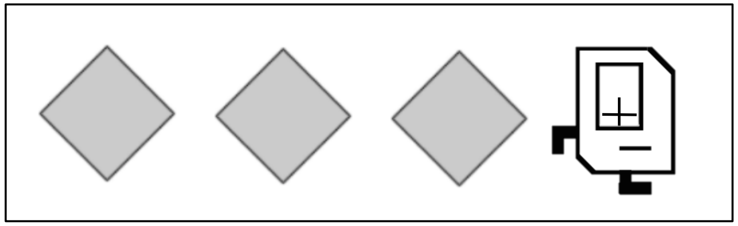


Place Beeper Line

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

Not in **while** loop

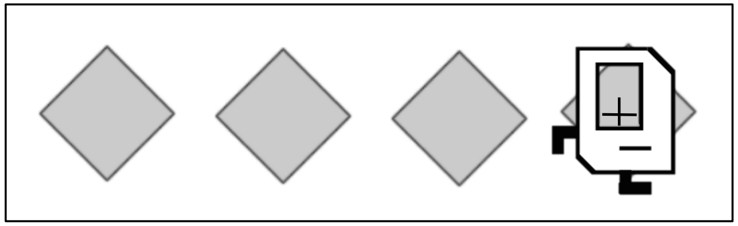
Fixed!



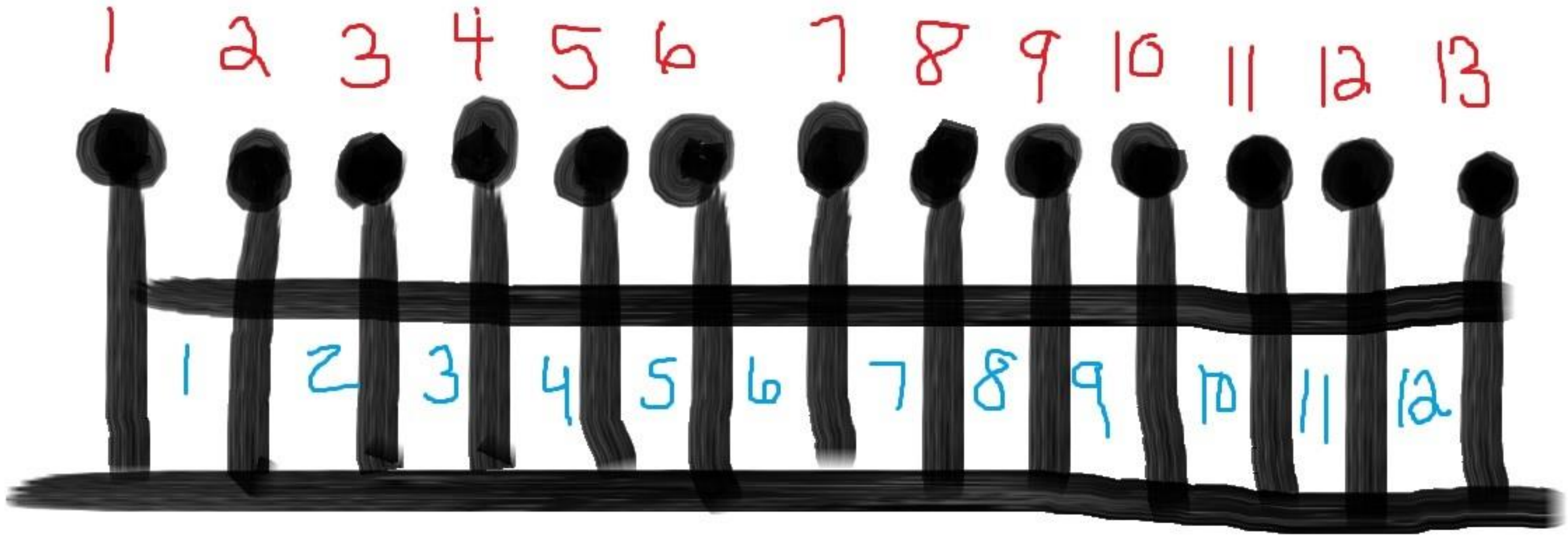
Place Beeper Line

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

Fixed!



Fence Post Problem

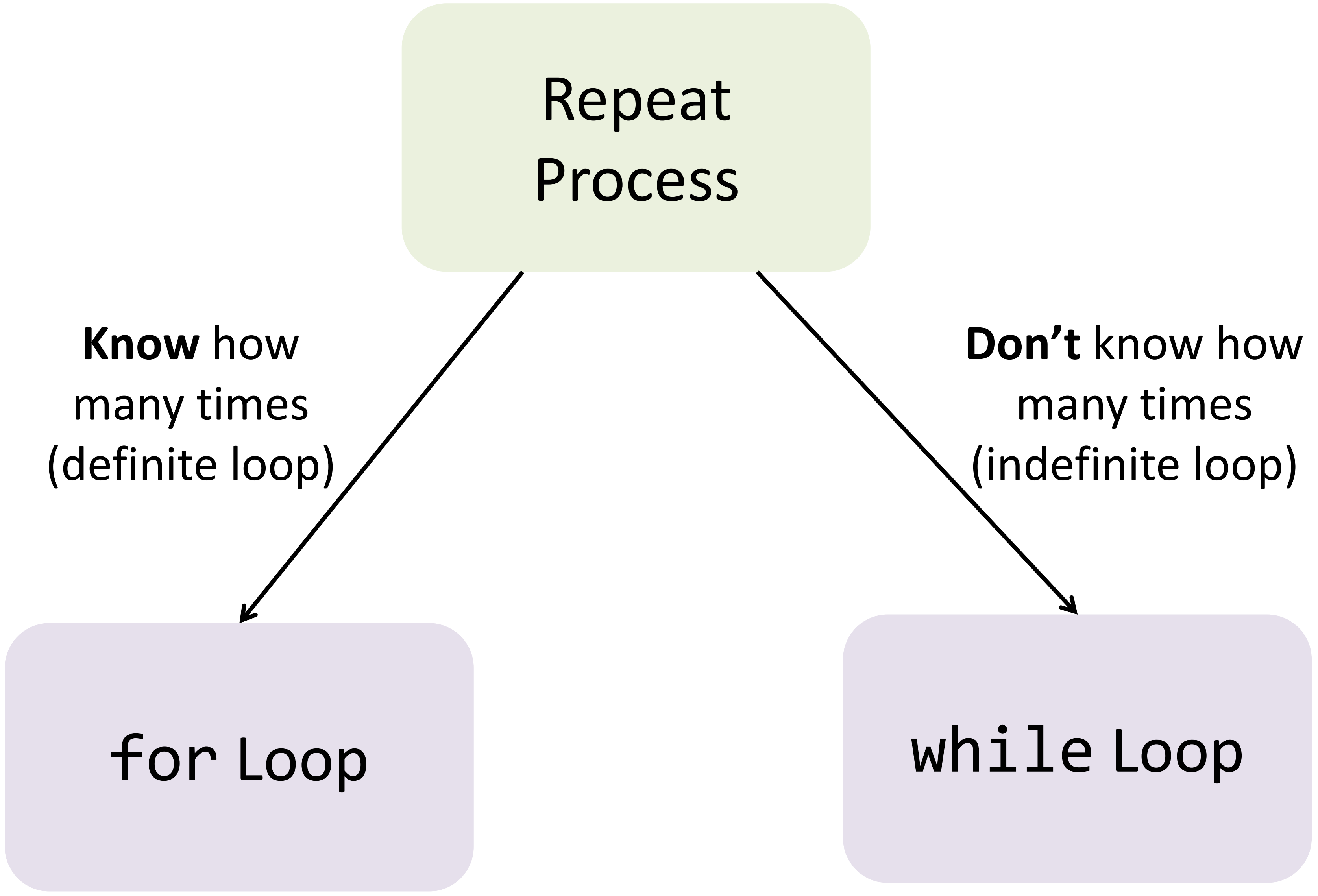


Also sometimes called an “Off By One Bug”

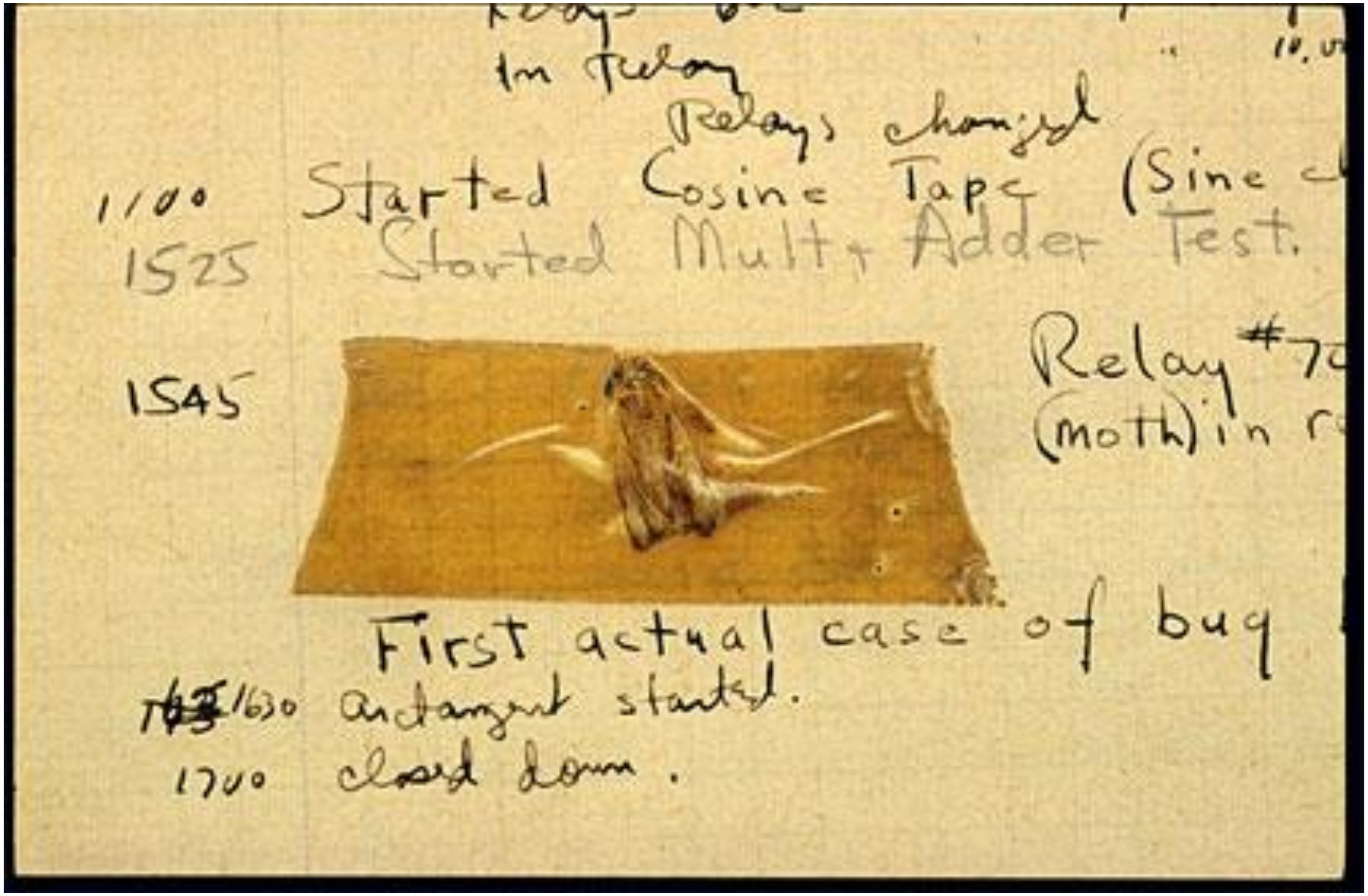
A program executes one line at a time.

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

Which Loop


Repeat Process
Know how many times (definite loop)
Don't know how many times (indefinite loop)
`for` Loop
`while` Loop

Actual Bug from Marc II



in relay

Relays changed

1100 Started Cosine Tape (Sine

1525 Started Mult + Adder Test.

1545 Relay #70

(moth) in r

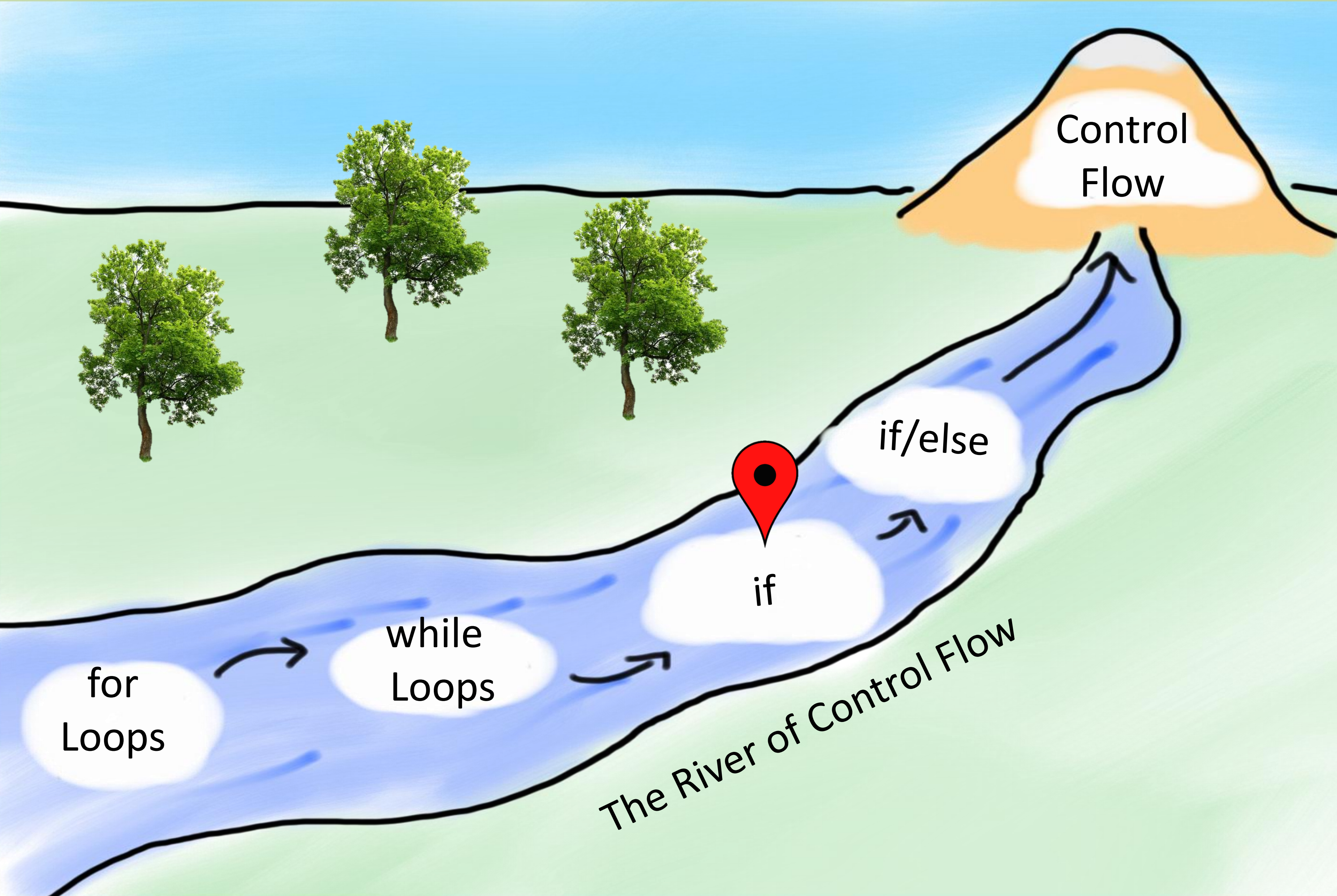
First actual case of bug

~~143~~ 1630 arctangent started.

1700 closed down.

Grace Hopper

Today's Route

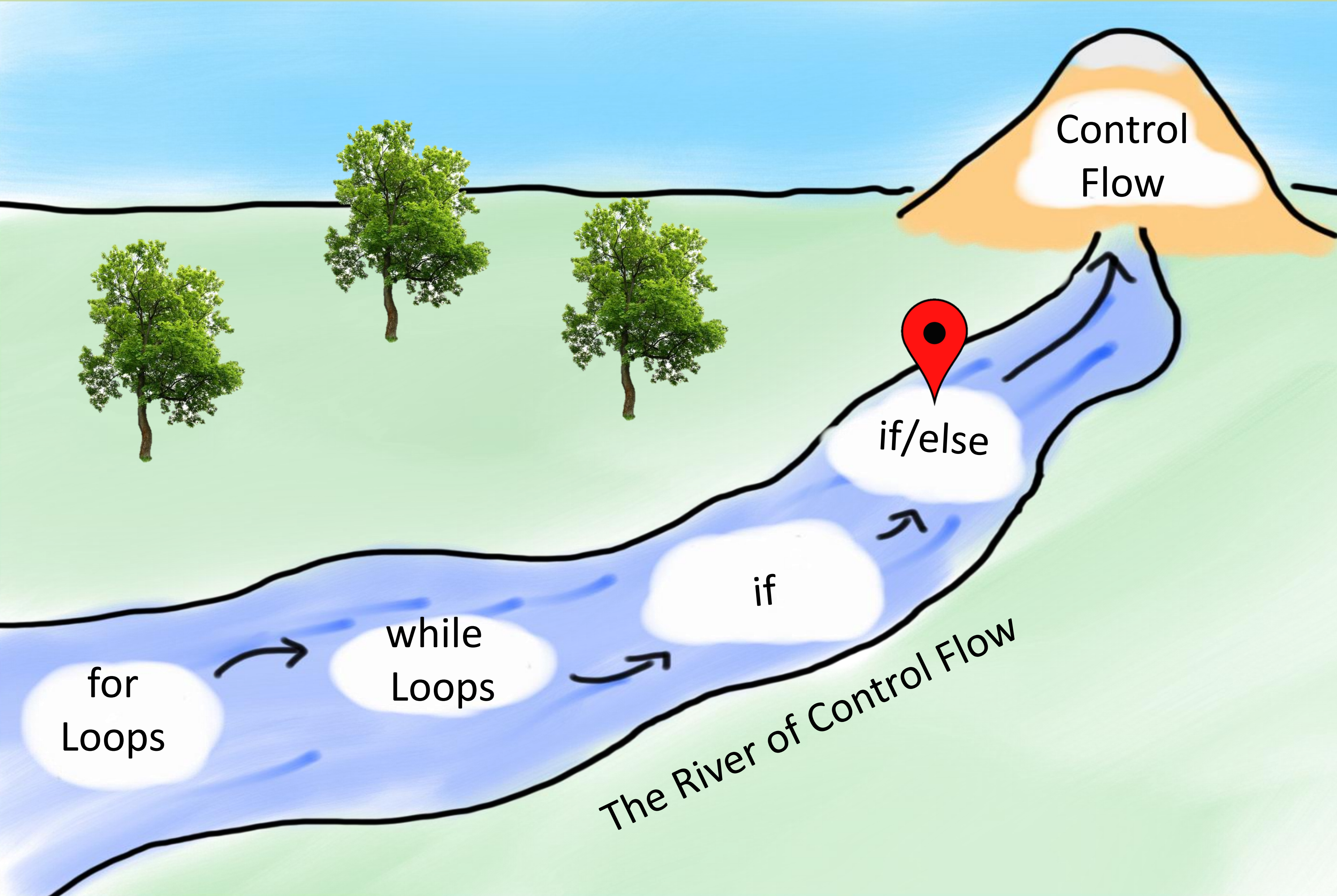


`if` statement

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

You just learned most of programming “control flow”

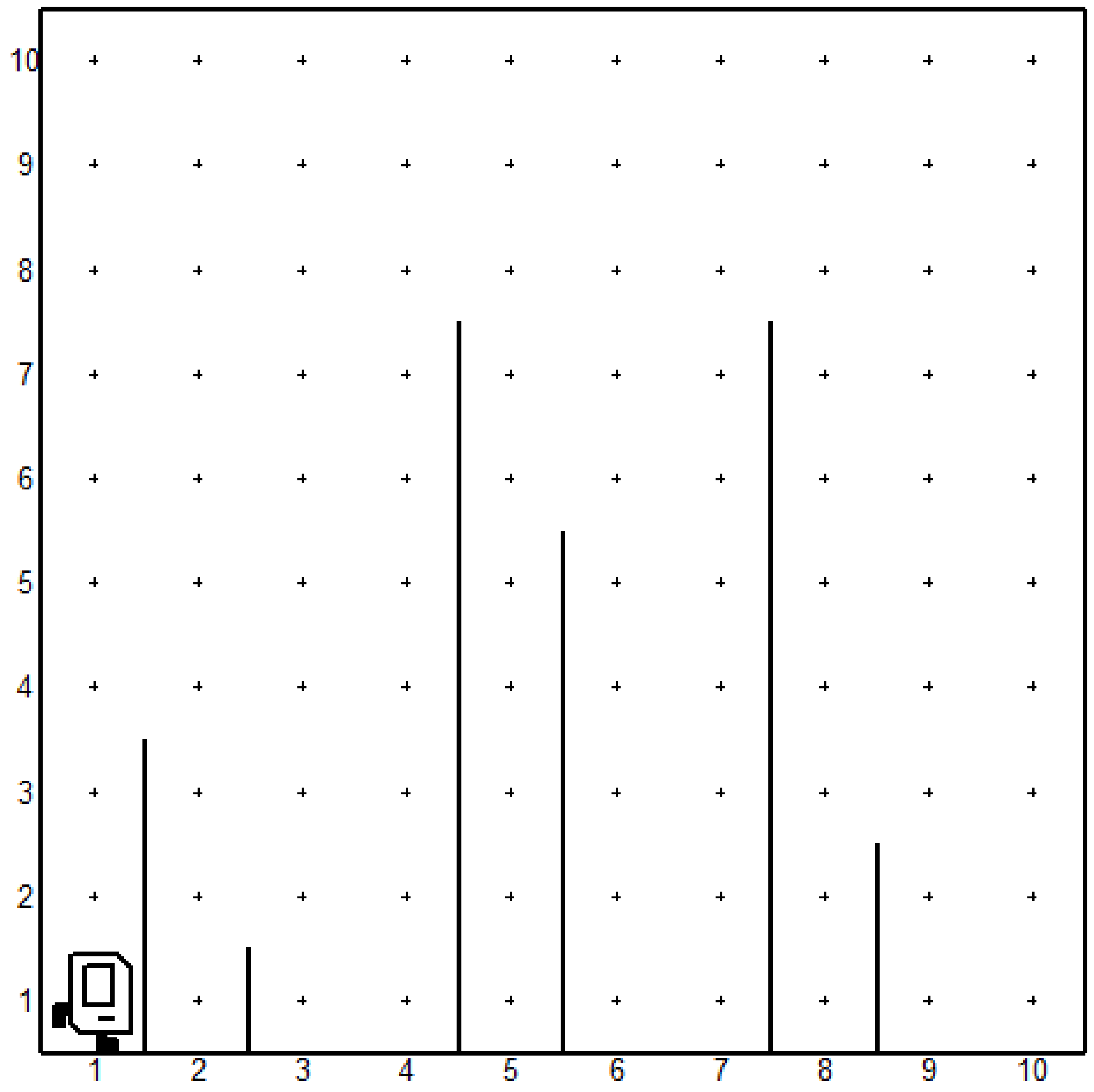
Today's Goal

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

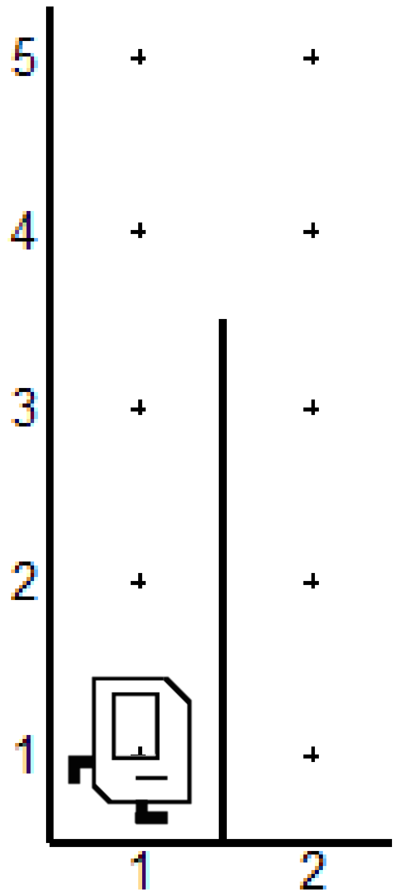
🎉 🙌 🙌

Putting it all together
SteepChaseKarel.py

Steeple Chase

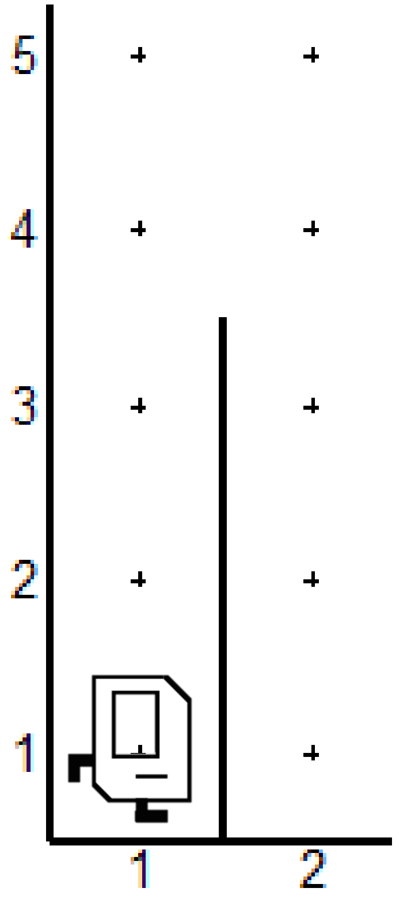


Focus on One Steeple



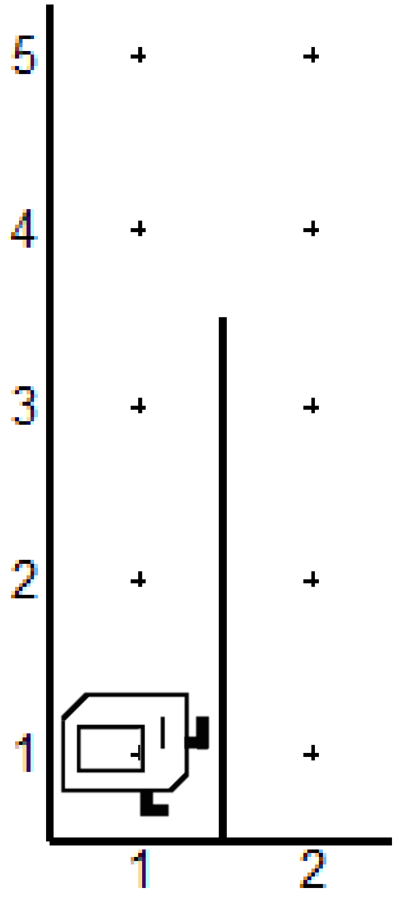
Focus on One Steeple

```
turn_left()
```

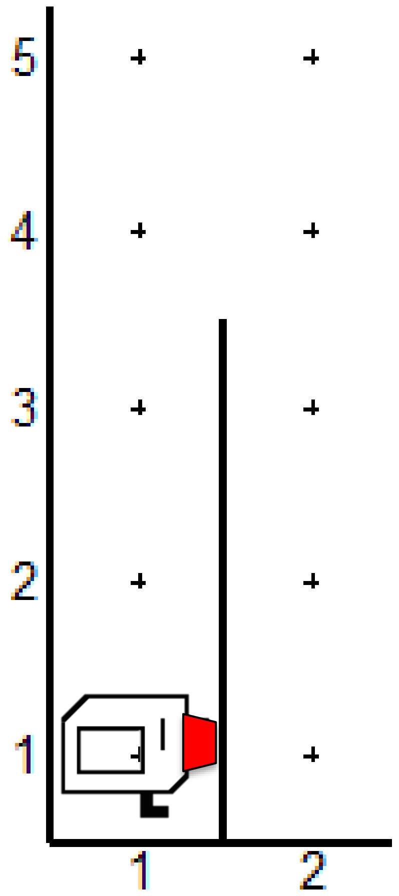


Focus on One Steeple

```
turn_left()
```

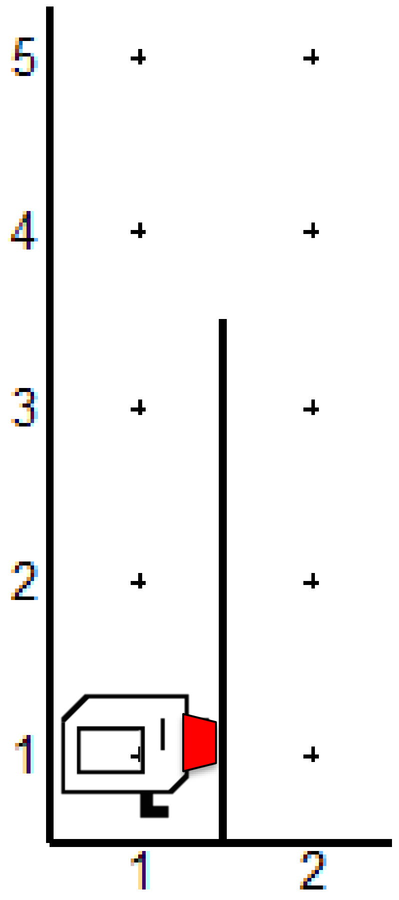


Focus on One Steeple



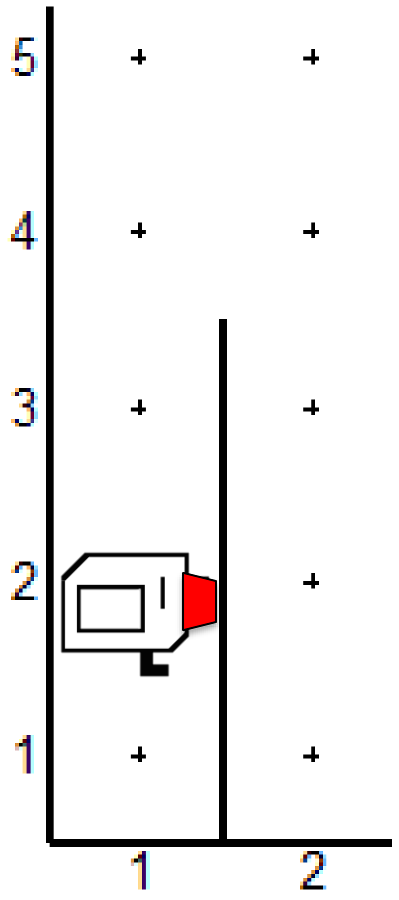
```
turn_left()
while right_is_blocked():
    move()
```

Focus on One Steeple



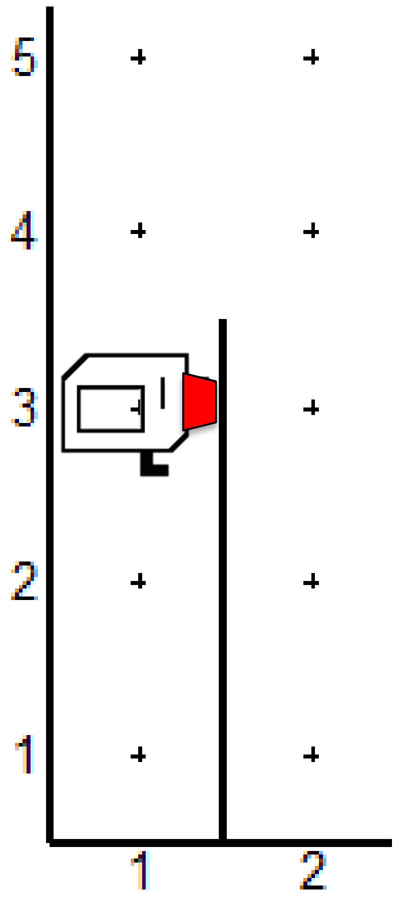
```
turn_left()
while right_is_blocked():
    move()
```

Focus on One Steeple



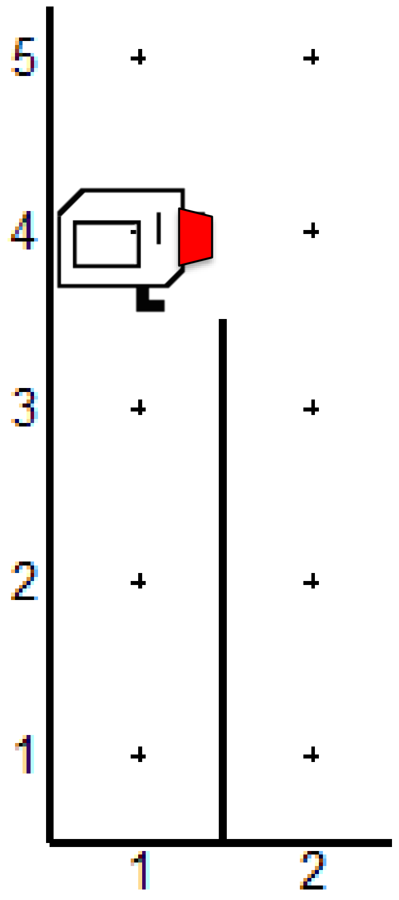
```
turn_left()
while right_is_blocked():
    move()
```

Focus on One Steeple



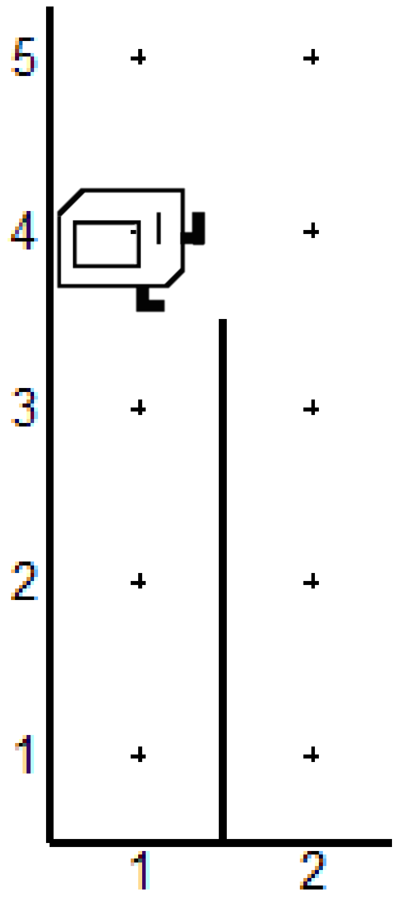
```
turn_left()
while right_is_blocked():
    move()
```

Focus on One Steeple



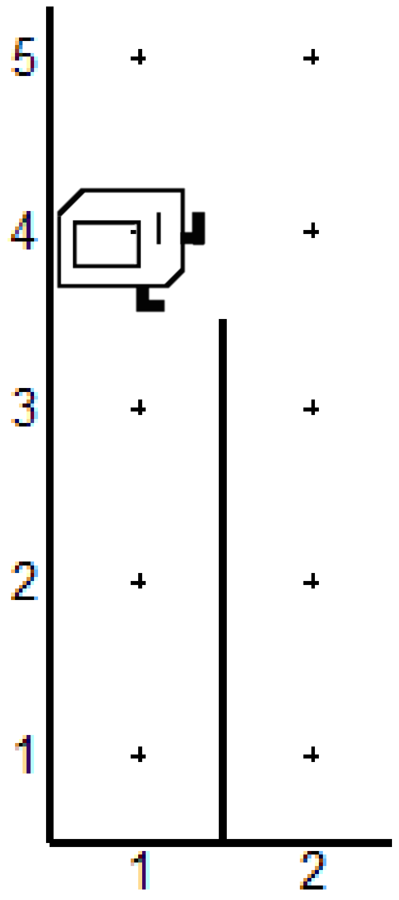
```
turn_left()
while right_is_blocked():
    move()
```

Focus on One Steeple



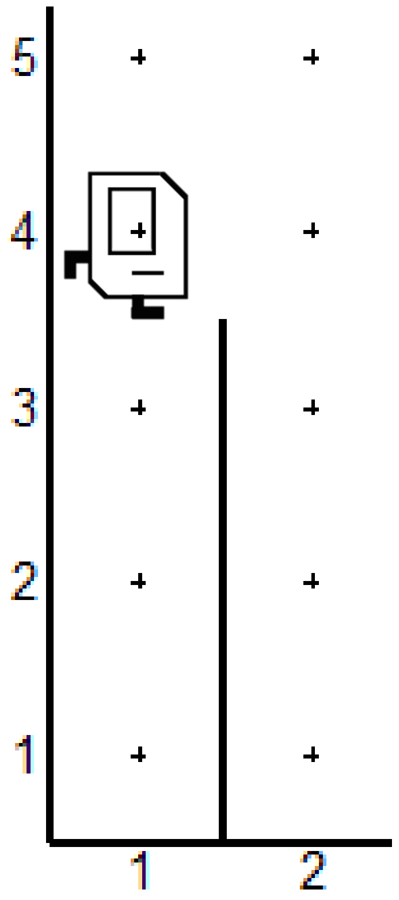
```
turn_left()
while right_is_blocked():
    move()
```

Focus on One Steeple



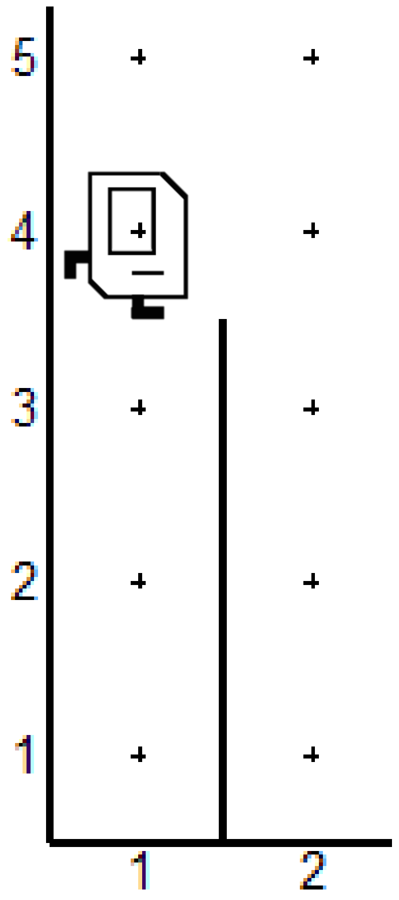
```
turn_left()
while right_is_blocked():
    move()
turn_right()
```

Focus on One Steeple



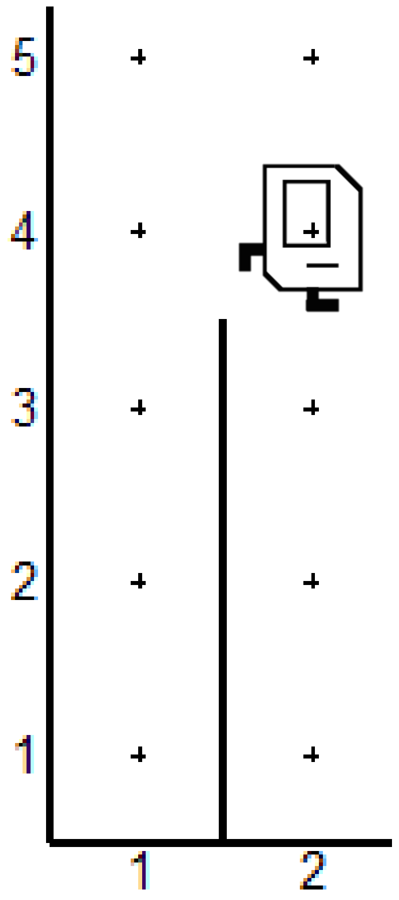
```
turn_left()
while right_is_blocked():
    move()
turn_right()
```

Focus on One Steeple



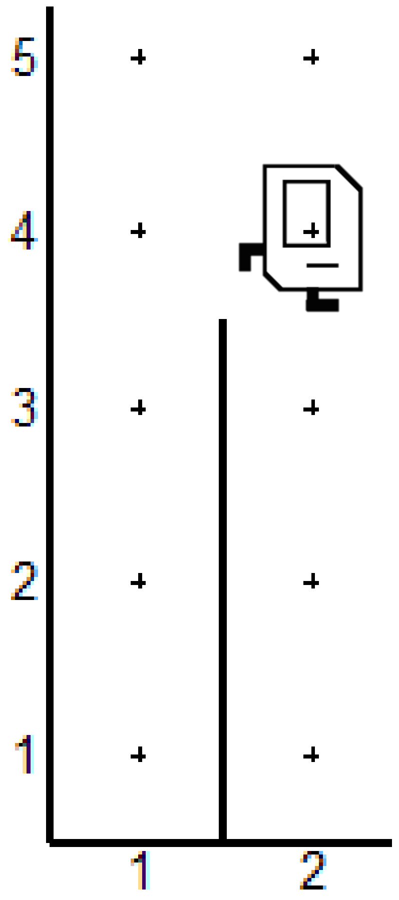
```
turn_left()
while right_is_blocked():
    move()
turn_right()
move()
```

Focus on One Steeple



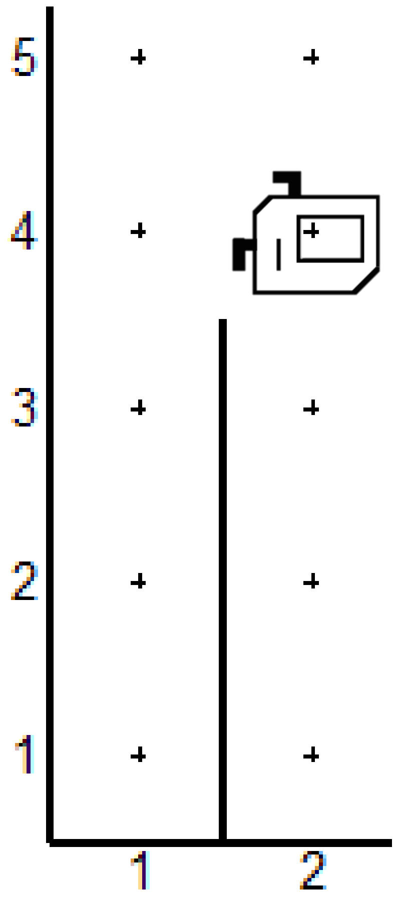
```
turn_left()
while right_is_blocked():
    move()
turn_right()
move()
```

Focus on One Steeple



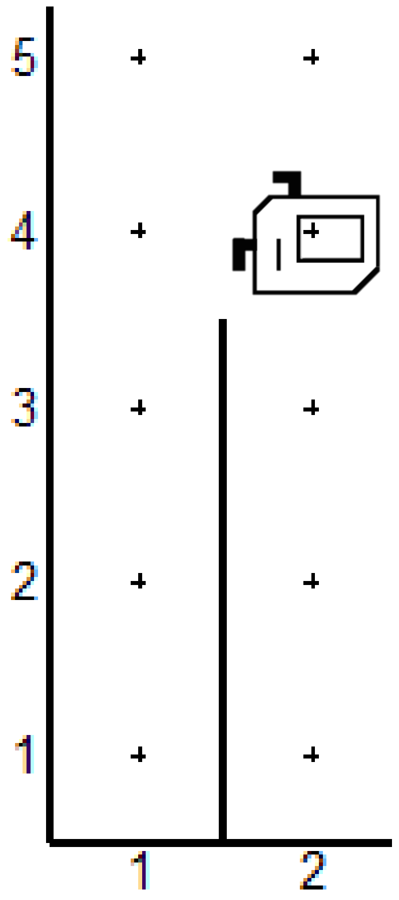
```
turn_left()
while right_is_blocked():
    move()
turn_right()
move()
turn_right()
```

Focus on One Steeple



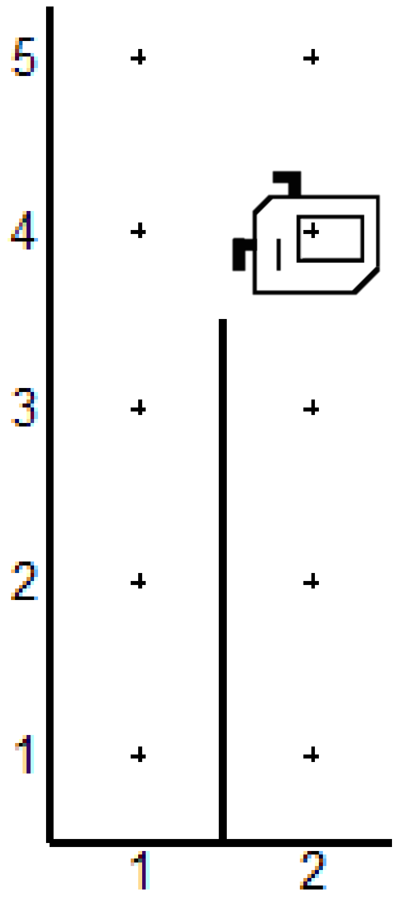
```
turn_left()
while right_is_blocked():
    move()
turn_right()
move()
turn_right()
```

Focus on One Steeple



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

Focus on One Steeple

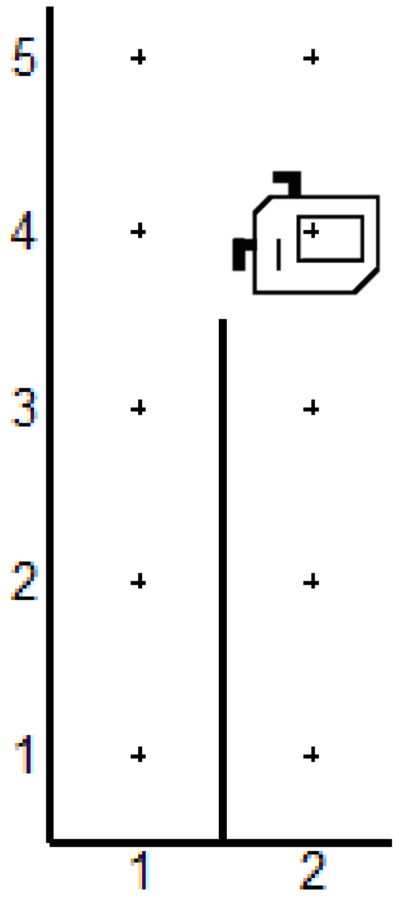


```
turn_left()
while right_is_blocked():
    move()
turn_right()
move()
turn_right()
move_to_wall()




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

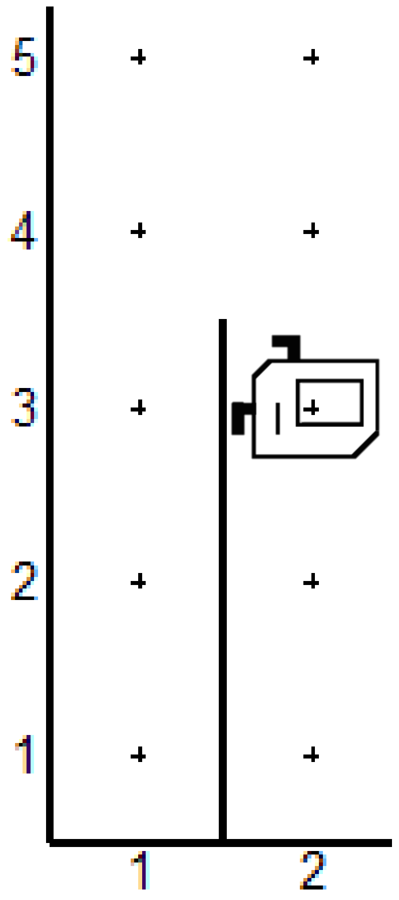
Focus on One Steeple



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

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

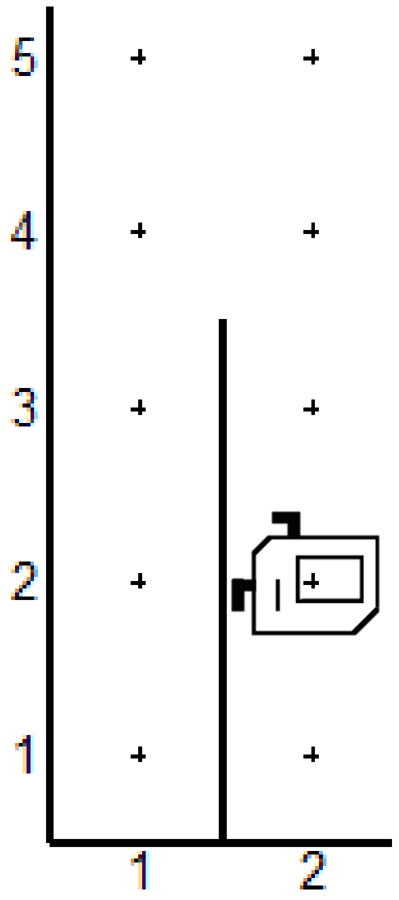
Focus on One Steeple



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

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

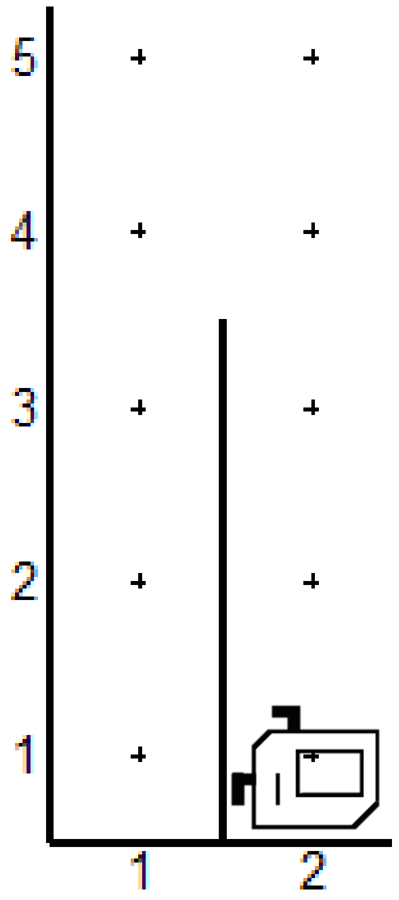
Focus on One Steeple



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

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

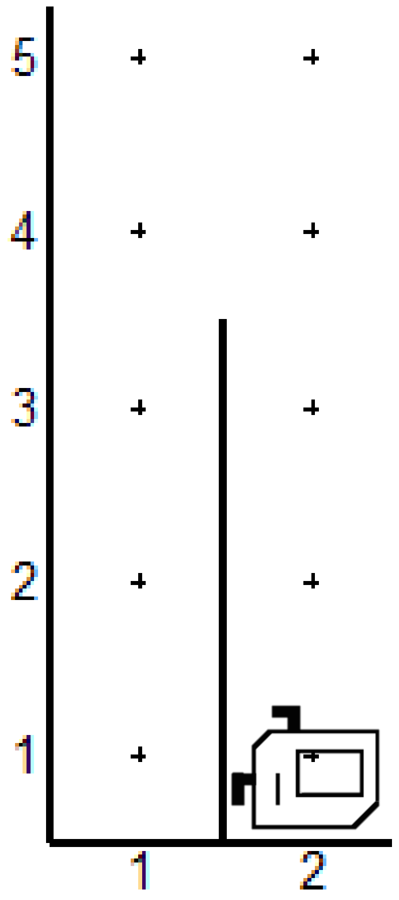
Focus on One Steeple



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

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

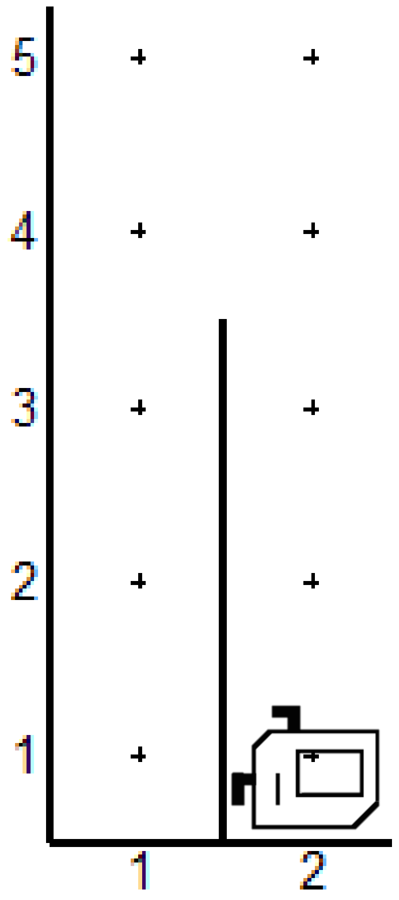
Focus on One Steeple



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

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

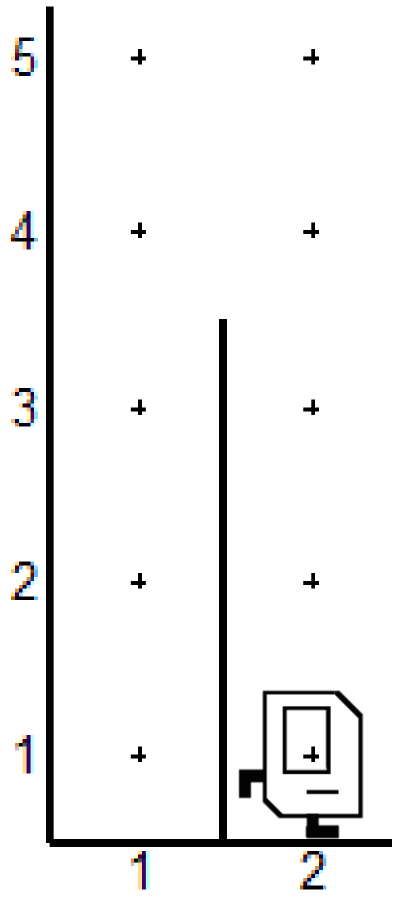
Focus on One Steeple



```
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

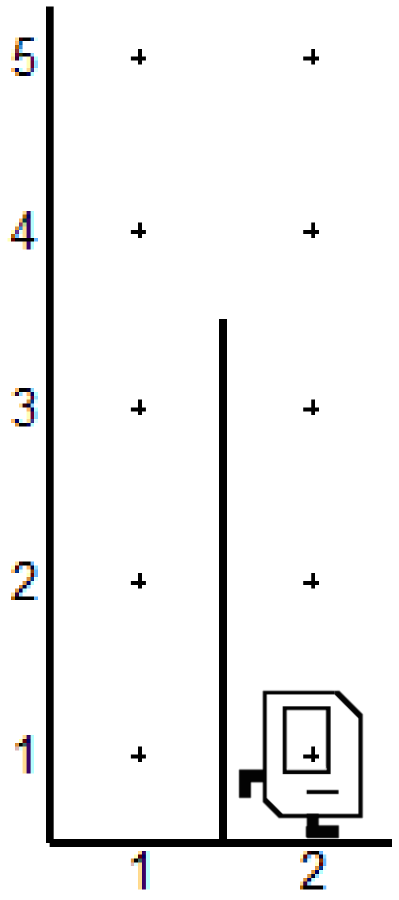


```
turn_left()
while right_is_blocked():
    move()
turn_right()
move()
turn_right()
move_to_wall()
turn_left()



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

Focus on One Steeple

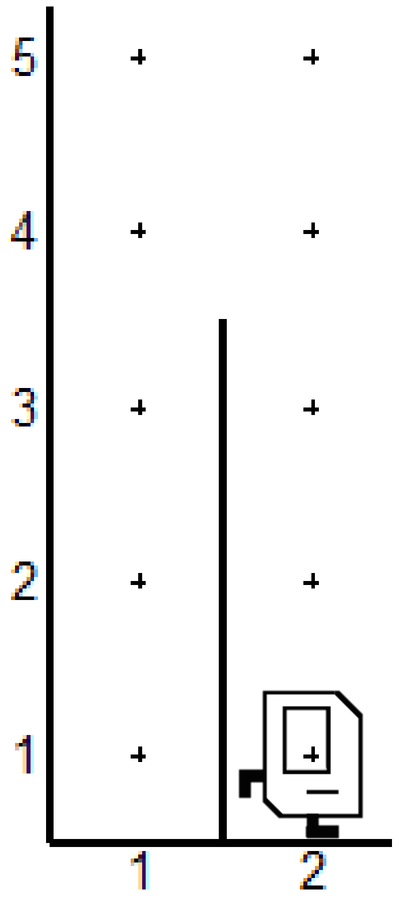


```
turn_left()
while right_is_blocked():
    move()
turn_right()
move()
turn_right()
move_to_wall()
turn_left()
```

You want 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