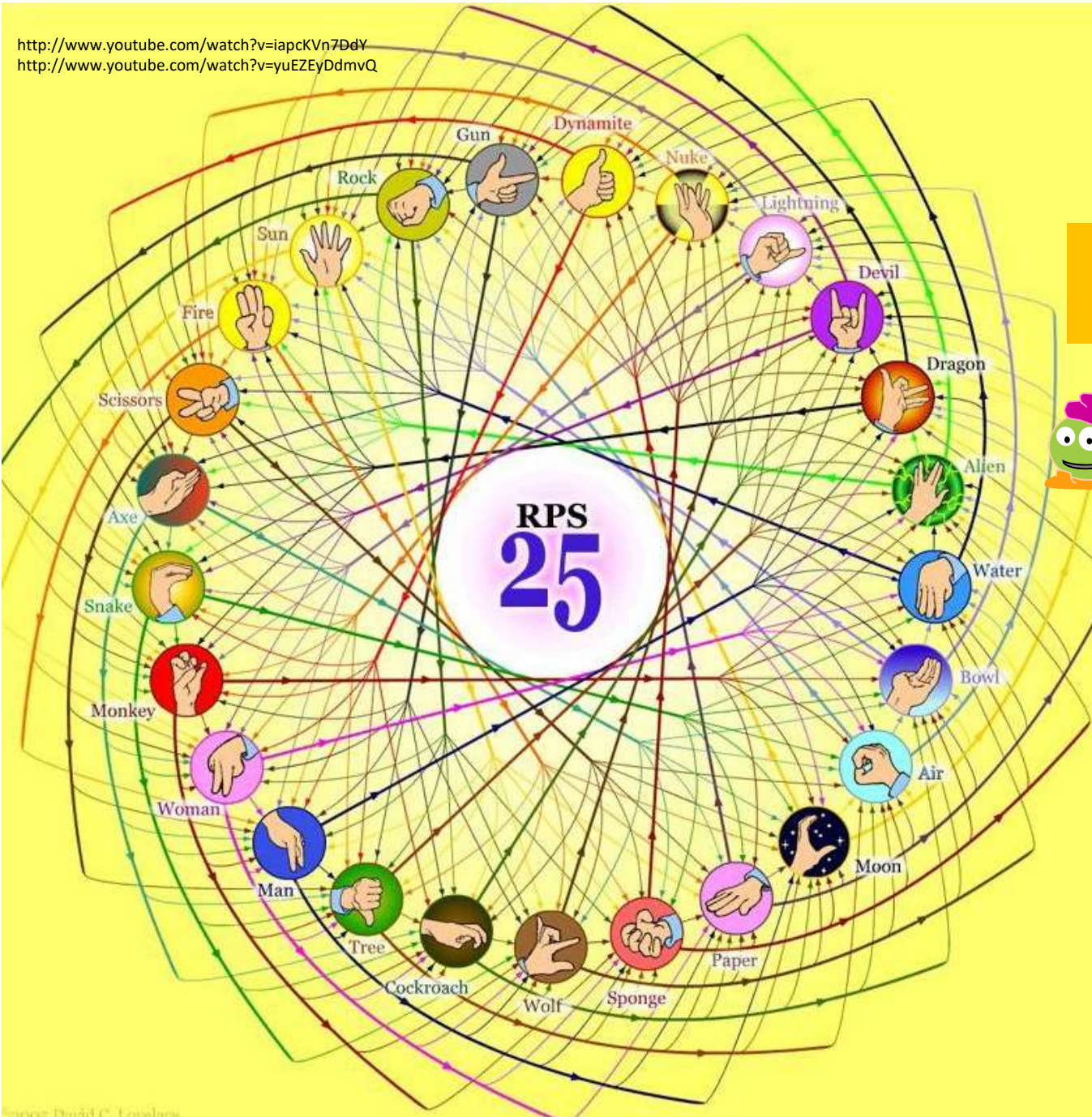


<http://www.youtube.com/watch?v=iapcKvN7DdY>
<http://www.youtube.com/watch?v=yuEZEyDdmvQ>



Source: David C. Lovelace

RPS-25 ?!



They call *that* an alien?

Spock *mind-melds* three-eyed aliens!



Provably.

Lab lookback...

Lab's goal: Get things working
Complete 25-50% of the hw

Nick's rule...

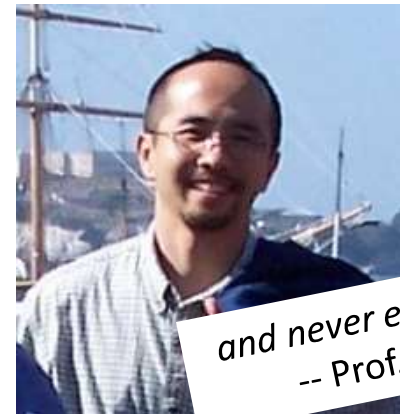
Finished with lab? OK! No *need* to stay longer



```
print "Thirty Three is", sqrt(4)/.4 + factorial(4) + 4  
print "Victory!"
```

Wow!?

Four fours is ~
sometimes too many...
othertimes too few...



and never enough!
-- Prof. Su

F. Su

1	$\frac{4!}{4}$	34	$4! + \frac{4!}{4} + 4$	67	$\frac{4! + \sqrt{4}}{.4} + \sqrt{4}$
2	$\frac{4!}{4} + \frac{4!}{4}$	35	$4! + \frac{4!}{4}$	68	$4 \cdot 4 \cdot 4 + 4$
3	$\frac{4! + 4!}{4}$	36	$4! + 4(4) - 4$	69	$\frac{4! \sqrt{4} - \sqrt{4}}{.4}$
4	$4! - 4(4) - 4$	37	$4! + \frac{4!}{4} + 4$	70	$4! \cdot 4 - 4! - \sqrt{4}$
5	$\frac{4!}{4} - \frac{4!}{4}$	38	$44 - \frac{4!}{4}$	71	$(4! + 4 \cdot 4) / .4$
6	$4 + \frac{4! + 4!}{4}$	39	$4! + 4! - \frac{4!}{4}$	72	$4^{4\sqrt{4}} / .4$
7	$\frac{4!}{4} + \frac{4!}{4}$	40	$4 \cdot 4 \cdot 4 - 4!$	73	$(4! \sqrt{4} + \sqrt{4}) / \sqrt{4}$
8	$4 + \frac{4(4)}{4}$	41	$44 - \sqrt{\frac{4!}{4}}$	74	$4! \cdot 4 - 4! + \sqrt{4}$
9	$4 + 4 + \frac{4!}{4}$	42	$44 - \frac{4!}{4}$	75	$(4! + 4 + \sqrt{4}) / .4$
10	$4(4) - \frac{4!}{4}$	43	$44 - \frac{4!}{4}$	76	$4! \cdot 4 - 4! + 4$
11	$(4! + 4) / 4 + 4$	44	$44 / \frac{4!}{4}$	77	$(\frac{4!}{4})^{\sqrt{4}} - 4$
12	$4! - (4 + 4 + 4)$	45	$44 + \frac{4!}{4}$	78	$(4! - 4) \cdot 4 - \sqrt{4}$
13	$4! - \frac{4!}{4}$	46	$44 + \frac{4!}{4}$	79	$(\frac{4!}{4})^{\sqrt{4}} - \sqrt{4}$
14	$\frac{4!}{4} + 4 + 4$	47	$4! + 4! - \frac{4!}{4}$	80	$4! \cdot 4 - 4(4)$

15	$4(4) - \frac{4}{4}$	48	$(4! + 4!) / \frac{4}{4}$	81	$(\frac{4!4!}{4})^{\frac{4}{4}}$
16	$4(4) / \frac{4}{4}$	49	$4! + 4! + \frac{4}{4}$	82	$(4! - 4) \cdot 4 + \sqrt{4}$
17	$4(4) + \frac{4}{4}$	50	$4! + 4! + \frac{4}{\sqrt{4}}$	83	$(\frac{4}{4})^{\frac{4}{4}} + \sqrt{4}$
18	$(4 \cdot 4) \frac{4}{4} + 4$	51	$4! + 4! + \sqrt{\frac{4}{4}}$	84	$4 \cdot 4 \sqrt{4} - 4$
19	$4! - 4 - \frac{4}{4}$	52	$4! + 4! + \sqrt{4} + \sqrt{4}$	85	$(\frac{4}{4})^{\frac{4}{4}} + 4$
20	$(4 + \frac{4}{4}) \cdot 4$	53	$4 \cdot 4 + \frac{4}{4}$	86	$4 \cdot 4 \sqrt{4} - \sqrt{\frac{4}{4}}$
21	$4! - 4 + \frac{4}{4}$	54	$4! + 4! + 4 + \sqrt{4}$	87	$4! \cdot 4 - \frac{4}{4}$
22	$\frac{4}{4} + 4(4)$	55	$\frac{4!}{\sqrt{4}} / 4$	88	$4! \cdot 4 - 4 - 4$
23	$4! - \sqrt{4} + \frac{4}{4}$	56	$4! + 4! + 4 + 4$	89	$\frac{4! + \sqrt{4}}{4} + 4!$
24	$4(4) + 4 + 4$	57	$4! + 4! + \frac{4}{4}$	90	$4! \cdot 4 - 4 - \sqrt{4}$
25	$4! + \sqrt{4} - \frac{4}{4}$	58	$(4! - 4 - 4) / 4$	91	$4! \cdot 4 - \frac{\sqrt{4}}{4}$
26	$4! + \frac{4! \cdot 4}{4}$	59	$[(\sqrt{4} \sqrt{4})! - 4] / 4$	92	$4! \cdot 4 - \sqrt{4} \sqrt{4}$
27	$4! + 4 - \frac{4}{4}$	60	$4 \cdot 4 \cdot 4 - 4$	93	$4! \cdot 4 - \sqrt{\frac{4}{4}}$
28	$4 \cdot 4 - 4(4)$	61	$[(\sqrt{4} \sqrt{4})! + 4] / 4$	94	$4! \cdot 4 - \frac{4}{\sqrt{4}}$
29	$4! + 4 + \frac{4}{4}$	62	$4 \cdot 4 \cdot 4 - \sqrt{4}$	95	$4! \cdot 4 - \frac{4}{4}$
30	$4! + \sqrt{4} \sqrt{4} + \sqrt{4}$	63	$(4^4 - 4) / 4$	96	$4! \cdot 4 / \frac{4}{4}$
31	$4! + \frac{4! + 4}{4}$	64	$4 \cdot 4 \sqrt{4} \cdot \sqrt{4}$	97	$4! \cdot 4 + \frac{4}{4}$
32	$4! + \sqrt{4} \sqrt{4} + 4$	65	$(4^4 + 4) / 4$	98	$4! \cdot 4 + \frac{4}{\sqrt{4}}$
33	$4! + \frac{\sqrt{4} \sqrt{4}}{4}$	66	$4 \cdot 4 \cdot 4 + \sqrt{4}$	99	$4! \cdot 4 + \sqrt{\frac{4}{4}}$
				100	$4! \cdot 4 + \sqrt{4} \sqrt{4}$

Email help: *Start w/ Piazza...*

for many questions, **Piazza** is a great resource:

this link:

Administration	Course Syllabus	Exam	Pairs Policy	Submission Info
Using Python	In your browser	On your machine	CS5 text	HTTLACS text
Useful/Help	Submission site	CS5 Piazza	<i>Grutoring!</i>	Picobot

this Q&A page

The screenshot shows the Piazza interface for a class. The top navigation bar includes 'CS5', 'Q & A', 'Resources', 'Statistics', and 'Manage Class'. The 'Q & A' tab is selected. Below the navigation bar, there are tabs for 'polls', 'hw1', 'hw2', 'hw3', and 'hw4'. The main content area is titled 'Welcome to CS5's piazza site...' and contains a message from the instructor. The interface includes a 'New Post' button, a search bar, and a rich text editor with various formatting options like bold, italic, and link.

In-person help: "*grutoring*"

every day there are tutoring hours in the **LAC** lab

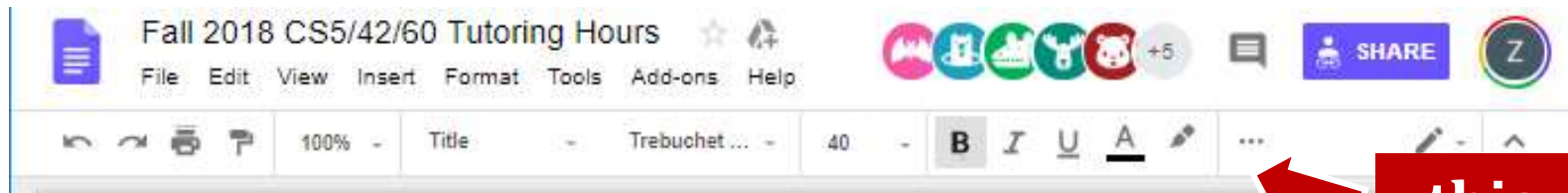
many days there are tutoring hours at other campuses

Linde
Activities
Center

this link:

Administration	Course Syllabus	Exams	Pairing	Info
Using Python	In your browser	On your machine	Grutoring!	...
Useful/Help	Submission site	CS5 Piazza	Grutoring!	Picobot

this link



this page

Friday in the LAC

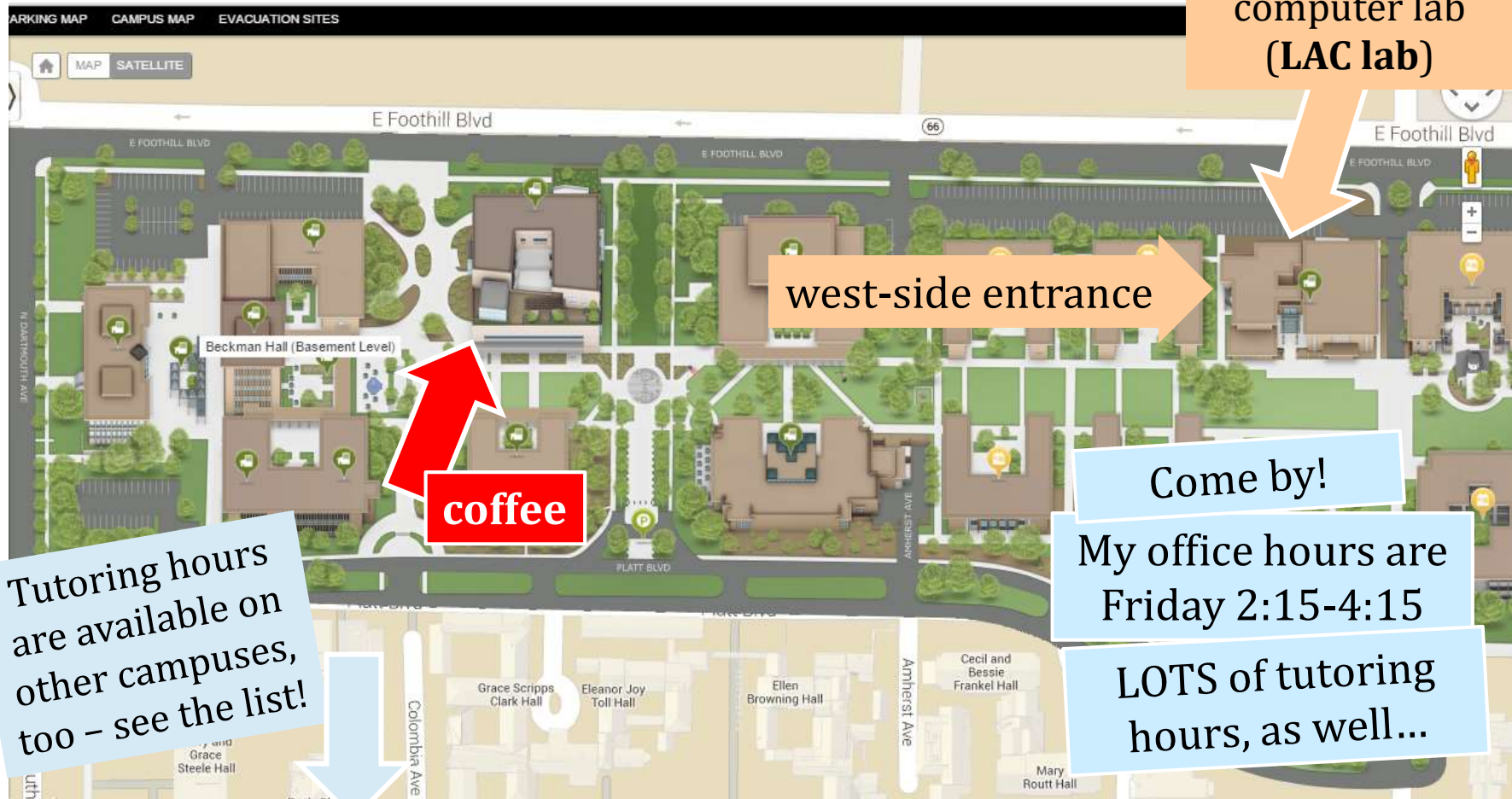
Friday 1pm-3pm

CS5 at HMC's LAC (up to 3)	No CS5 off-campus tutoring on Fridays	CS 42 (up to 1)	CS 60 (up to 1)
Serenity Wade		Sarah Embry <sembry@hmc.edu>	Seth Isaacson (sisaacson@hmc.edu)
Sydney Wallace <smwallace@g.hmc.edu>			
Yeabtsega (Sega) Birhane < ybirhane@g.hmc.edu > I'm			
Zach Dodds (2:15-4:15 or so) < zdodds@gmail.com >			

DON'T sign up JUST GO!
Grutors there to provide support...

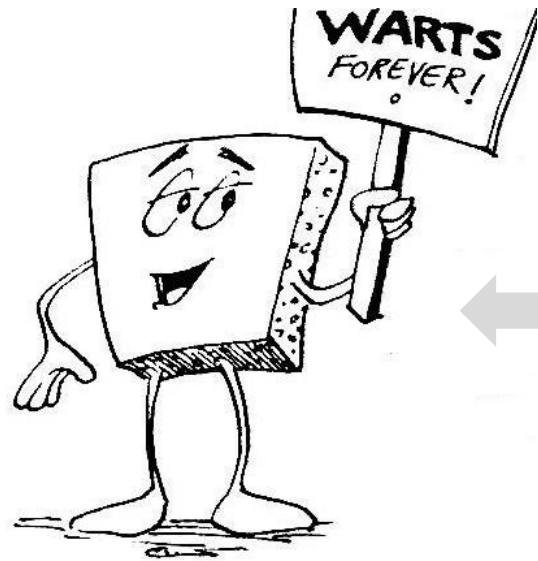
Tutoring location @ HMC: *LAC*

Tutoring hours are in the Linde Activities Center computer lab (**LAC lab**)



Tutoring hours are available on other campuses, too - see the list!

Welcome back to CS 5 !



Wally

Average of
these two?



Alien

Homework 0

due Mon. night (11:59pm)

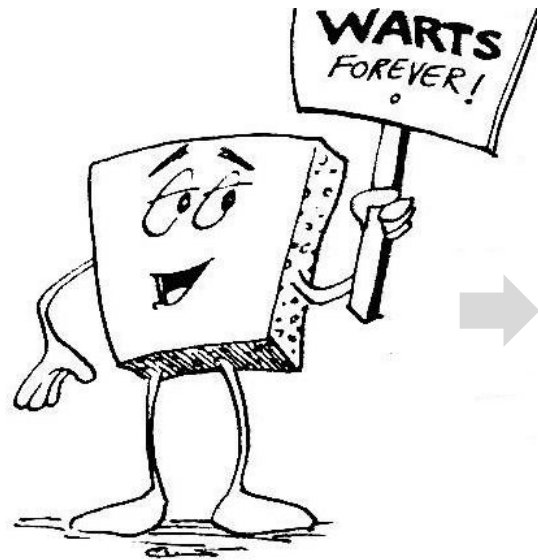
Problem 0: Reading + response...

Problem 1: Four-fours program: Can be done for lab...

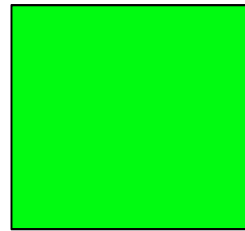
Problem 2: Rock-paper-scissors program (*Maybe* done already!)

Problems 3-4: Picobot! empty room (3) maze (4)

Welcome back to CS 5 !



Wally



Picobot!



Alien

Yes! I see the resemblance

Homework 0
due Mon. night (11:59pm)

Problem 0: Reading + response...

Problem 1: Four-fours program: Can be done for lab...

Problem 2: Rock-paper-scissors program (*Maybe* done already!)

Problems 3-4: Picobot! empty room (3) maze (4)

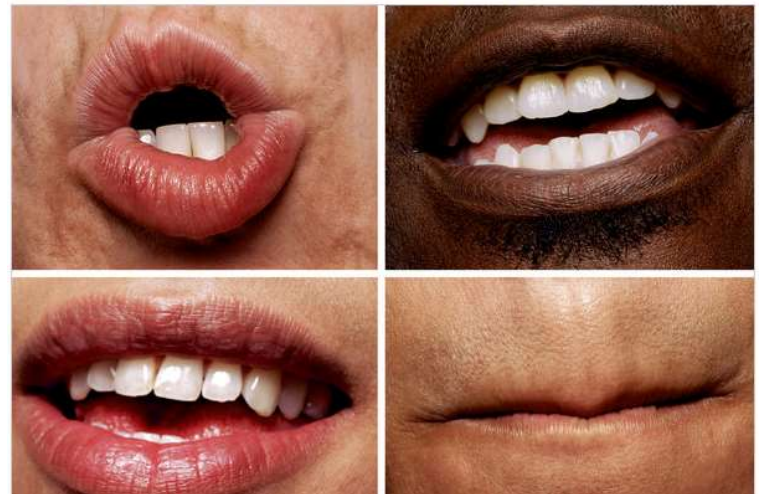
Problem 0 ?

Typically an article on CS or an application...

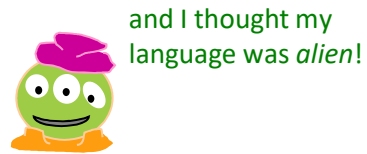
Submit a one-paragraph response { A few sentences that raise or address questions, using the article as a guide.

Small part (5 pts) {
5 - insightful, careful
4 - thoughtful
3 - complete, on topic
0-2 - less than complete

Does Your Language Shape How You Think?



This week's article might not seem like CS at first...



Does Your Language Shape How You Think?



Seventy years ago, in 1940, a popular science magazine published a short article that set in motion one of the trendiest intellectual fads of the 20th century. At first glance, there seemed little about the article to augur its subsequent celebrity. Neither the title, “Science and Linguistics,” nor the magazine, M.I.T.’s Technology Review, was most people’s idea of glamour. And the author, a chemical engineer who worked for an insurance company and moonlighted as an anthropology lecturer at Yale University, was an unlikely candidate for international superstardom. And yet Benjamin Lee Whorf let loose an alluring idea about language’s power over the mind, and his stirring prose seduced a whole generation into believing that our mother tongue restricts what we are able to think.

Seventy years ago, in 1940, a popular

But then a remote Australian aboriginal tongue, Guugu Yimithirr, from north Queensland, turned up, and with it came the astounding realization that not all languages conform to what we have always taken as simply “natural.” In fact, Guugu Yimithirr doesn’t make any use of egocentric coordinates at all. The anthropologist John Haviland and later the linguist Stephen Levinson have shown that Guugu Yimithirr does not use words like “left” or “right,” “in front of” or “behind,” to describe the position of objects. Whenever we would use the egocentric system, the Guugu Yimithirr rely on cardinal directions. If they want you to move over on the car seat to make room, they’ll say “move a bit to the east.” To tell you where exactly they left something in your house, they’ll say, “I left it on the southern edge of the western table.” Or they would warn you to “look out for that big ant just north of your foot.” Even when shown a film on television, they gave descriptions of it based on the orientation of the screen. If the television was facing north, and a man on the screen was approaching, they said that he was “coming northward.”



believing that our mother tongue restricts what we are able to think.

Last time...

What is CS?

CS is the study of **complexity**

How can **it** be done?

How well can **it** be done?

Can **it** be done at all?

CS's 6 big questions are here.

But only one is **programming**.
Do you see which?

Can you solve this problem? CS

Can you create a process to solve such problems?
programming + CS

How quickly can you find solutions? CS

Do you have the "best" solution? CS

Is every problem solvable? CS

Is there a way to tell?
There isn't always! CS

CS != Programming

What *is* programming ?

Programming as recipe-writing

vs.

Programming as learning a foreign language

1) Expect it to be different!

2) Don't memorize anything!

3) Immerse == Experiment!

Baggage!

ser/estar
go/went

**What about the *Python*
programming language ?**

Python ?



One possible relationship...

Python !



One possible relationship...



Happy co-existence...
It can even be comfy!

The *foreign language* of python...

syntax

How it looks

semantics

What it does

intent

What it should do



The *foreign language* of python...

syntax

How it looks

semantics

What it does

intent

What it should do



```
name = raw_input('Hi... what is your name? ')
print                                     # prints

if name == 'Eliot' or
    print 'T'

elif name
    print                                     , 'Oh.'

else:                                     # in all other cases...
    print 'welcome', name, '!'
    my_choice = random.choice( [ 'R', 'P', 'S' ] )
    print 'My favorite object is', my_choice, "!"
```

This program should greet its user appropriately.

The *foreign language* of python...

syntax

How it looks

semantics

What it does

intent

What it should do



```
name = raw_input('Hi... what is your name? ')
print                               # prints

if name == 'Eliot' or
    print 'T'

elif name
    print                               , 'Oh.'

else:                                # in all other cases
    print 'welcome', name, '!'
    my_choice = random.choice( [ 'R', 'P', 'S' ] )
    print 'My favorite object is', my_choice, "!"
```

**This program should greet
its user appropriately**

**human-
desired
output**

The *foreign language* of python...

syntax

How it looks

semantics

What it does

intent

What it should do



```
name = raw_input('Hi... what is your name? ')
print                                     # prints a blank line

if name == 'Eliot' or name == 'Ran':
    print 'I\'m "offline." Try later.'

elif name == 'Zach':                    # is it Zach?
    print 'Zach Quinto...?', 'No?', 'Oh.'

else:                                    # in all other cases...
    print 'Welcome', name, '!'
    my_choice = random.choice( [ 'R','P','S' ] )
    print 'My favorite object is', my_choice, "!"
```

The *foreign language* of python...

syntax

How it looks

semantics

What it does

intent

What it should do



```
name = raw_input('Hi... what is your name? ')
print                                     # prints a blank line

if name == 'Eliot' or name == 'Ran':
    print 'I\'m "offline." Try later.'

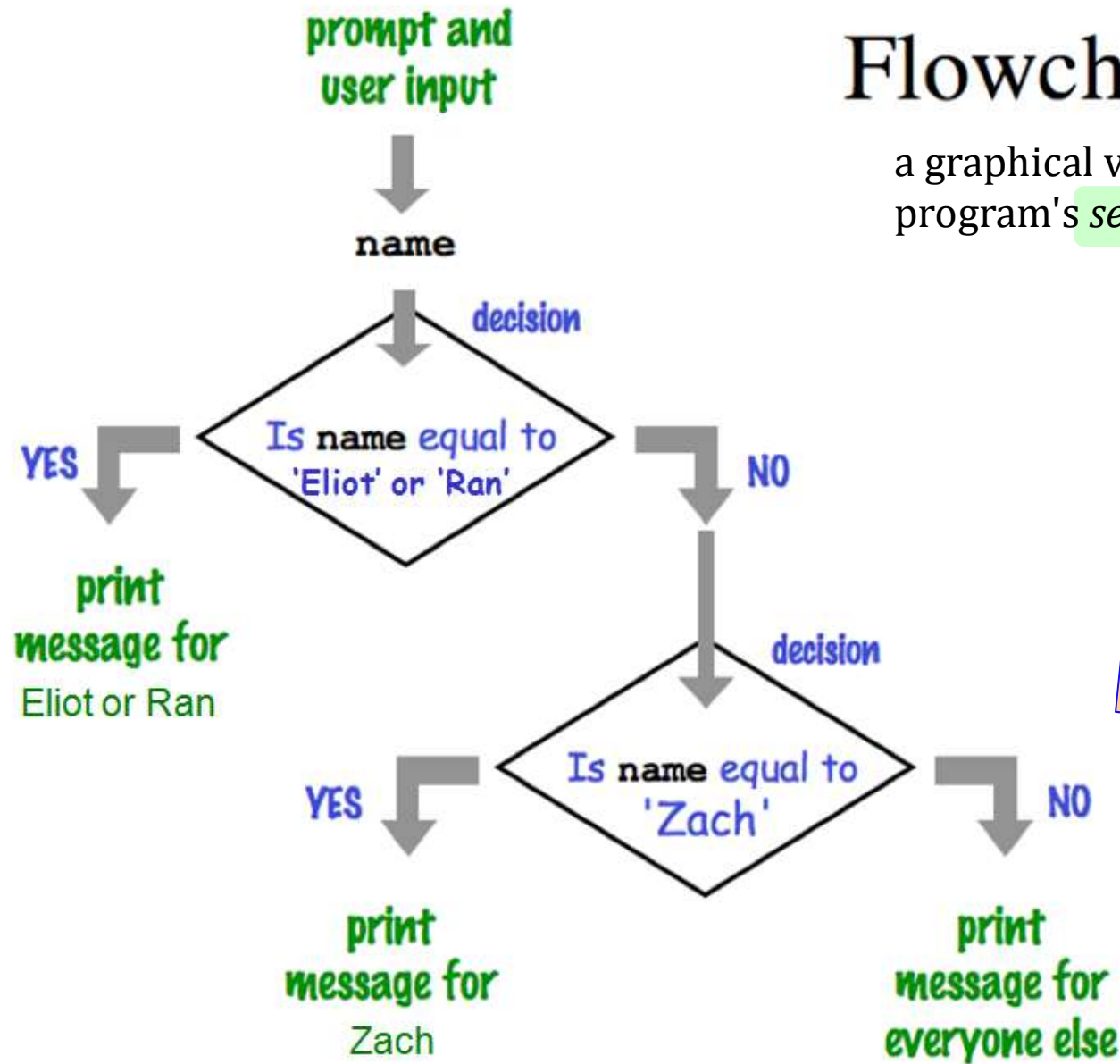
elif name == 'Zach':                    # is it Zach?
    print 'Zach Quinto...?', 'No?', 'Oh.'

else:                                    # in all other cases...
    print 'Welcome', name, '!'
    my_choice = random.choice( [ 'R','P','S' ] )
    print 'My favorite object is', my_choice, "!"
```

machine-
produced
output

Flowchart...

a graphical view of a program's *semantics*



machine-produced output



The *foreign language* of python...

syntax

How it looks



semantics

What it does

intent

What it should do



How
Python
looks!

- how punctuation is used
- the language keywords used
- use of whitespace
- peculiarities of formatting
- how behavior is affected ...

The *foreign language* of python...

syntax

How it looks



**human-
typed
input**

semantics

it does

intent

What it should do



How
Python
looks!

- how punctuation is used
- the language keywords used
- use of whitespace
- peculiarities of formatting
- how behavior is affected ...

The *challenge* of programming...

This is somehow familiar...?!



syntax

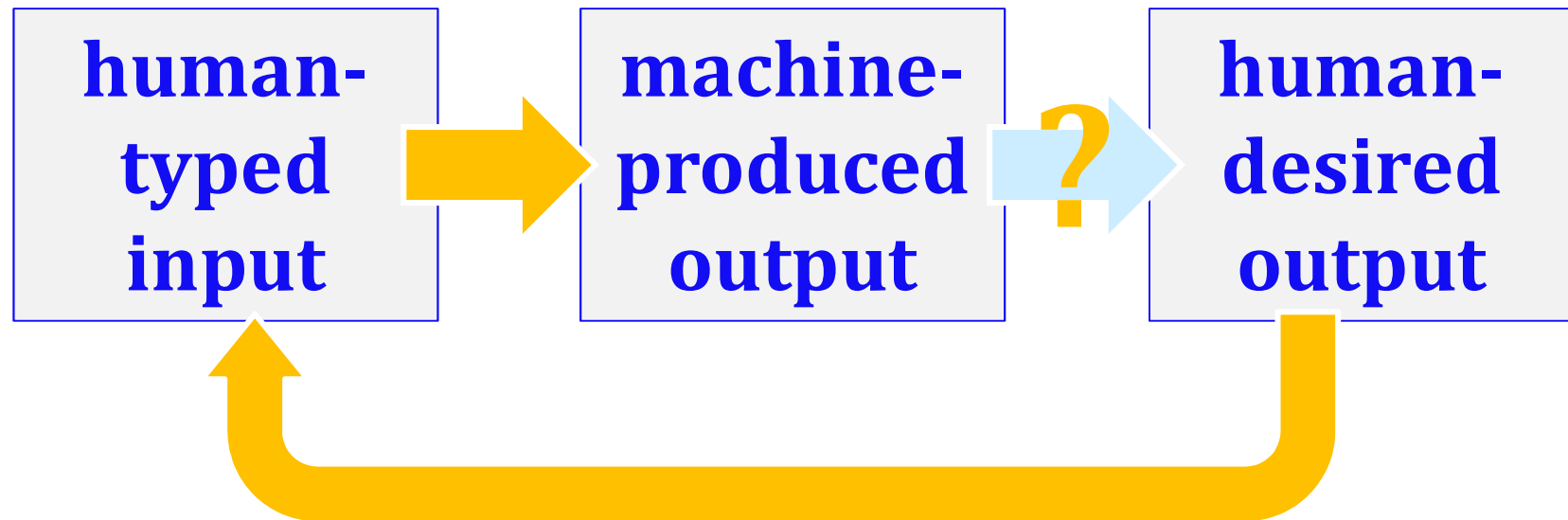
How it looks

semantics

What it does

intent

What it should do



The *challenge* of programming...

Look deep into my eyes...

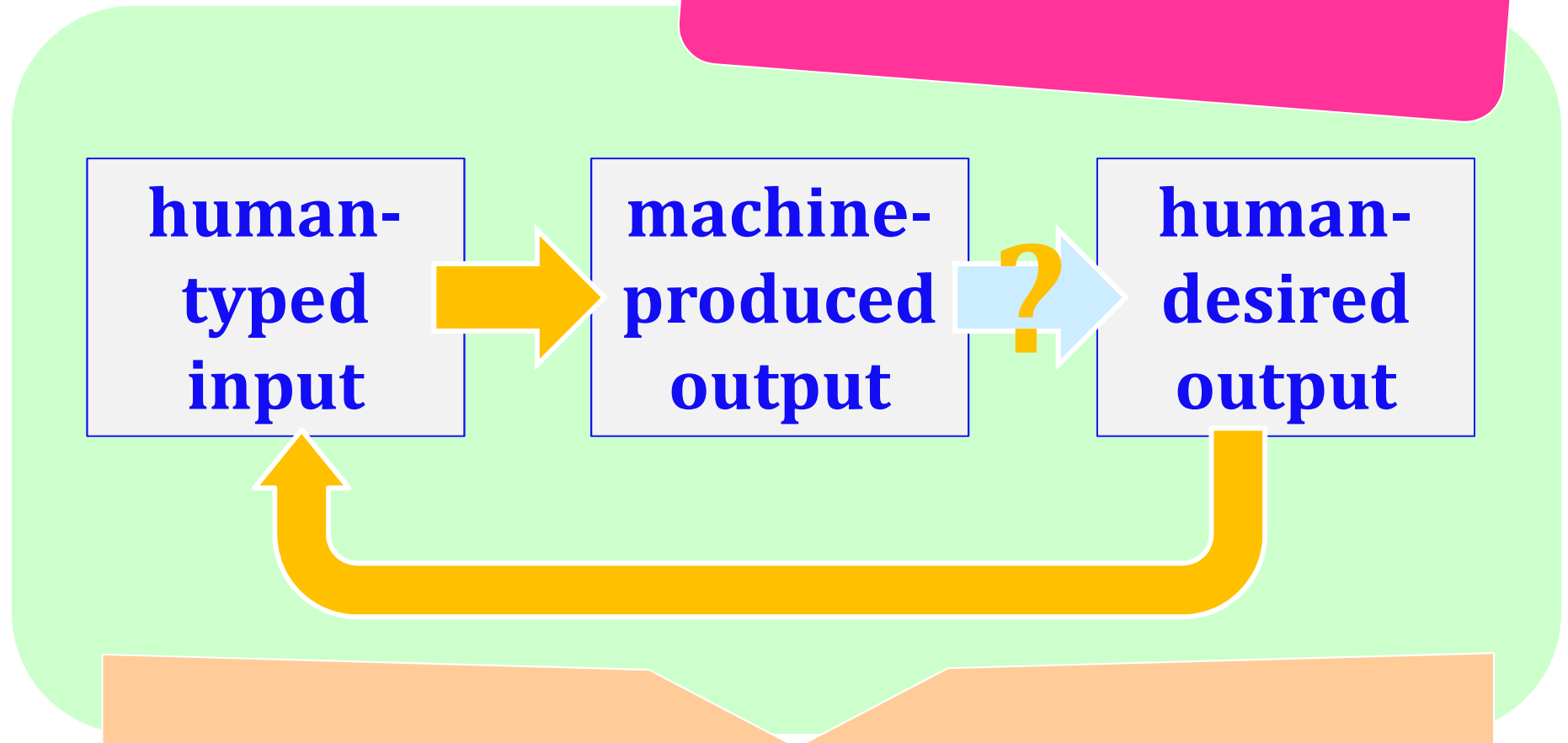


syntax

How it looks

semantics

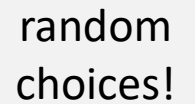
intent



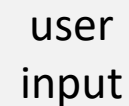
hw0pr2a: RPS...

```
1  #
2  # RPS example starting point
3
4  import random
5
6  print("Welcome to rock/paper/scissors, human!\n")
7
8  comp = random.choice(['rock', 'paper', 'scissors'])
9  user = input("  +++  Choose wisely: ")
10
11 print(" You chose", user)
12 print(" I chose", comp)
13 print()
14
15 if user == 'rock':
16     if comp == 'paper':
17         print(" paper defeats rock - I win!")
18
```

random
choices!



user
input



choices w/
if/elif/else



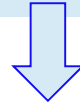
Name(s):

(1) Find and correct as many errors as you can in this code:

Syntax challenge!

```
import random
```

(2) This one line does *three* things... what are they?



```
user = input( "Choose your weapon! " )
```

```
comp = random.choice( ['rock', 'paper', 'scissors'] )
```

```
print('user (you) chose:', 'user')
```

```
print('comp (me!) chose:' comp)
```

```
if user == rock and comp = 'paper'
```

```
    print('The result is, YOU LOSE.'
```

```
    print('unless you're a CS 5 grader, then YOU WIN!')
```

(3) Extra! Can you find 7 punctuation marks used in *more than one way* here?

Syntax challenge!

(1) Find and correct as many errors as you can here...

(2) This line is doing *three* things... what are they?

every block of code must line up!

```
import random
```

set-equals always uses
ONE equals sign

```
user = input( "Choose your weapon! " )
```

```
comp = random.choice( ['rock', 'paper', 'scissors'] )
```

```
print('user (you) chose:', user)
```

```
print('comp (me!) chose:', comp)
```

test-equals uses TWO equals signs

The comma prints a space and
does NOT go to the next line.

match brackets
and quotes !

```
if user == rock and comp == 'paper':
```

a colon starts a new block

```
print('The result is, YOU LOSE.')
```

```
print('unless you\'re a CS 5 grader, then YOU WIN!')
```

a backslash handles special characters

flattering - or flouting -
graders is encouraged!

(3) Punctuation used in more than one way: () . ' = , :

Tear off that page

*Pass those to the aisles + **Eastward...***

be sure your name's on one...

Take a picture if you'd like to "keep" it

... then turn back into the packet

hw0pr2b: *Your Quest!*

```
1 # coding: utf-8
2 #
3 # Create a short text-
4 # adventure in Python...
5
6 """
7 Title for your adventure:  The Quest.
8
9 Notes on how to "win" or "lose" this adventure:
10     To win, choose the table.
11     To lose, choose the door.
12
13 """
14
15 import time
16
17 def adventure():
18     """ this function runs one session of interactive fiction
19         Well, it's "fiction," depending on the pill color chosen...
20         inputs: no inputs    (prompted text doesn't count as input)
21         outputs: no outputs  (printing doesn't count as output)
```

Use at least five control structures with decisions: (if/elif/else)

We look forward to adventuring!

Ln 17, Col 14 (9 selected) Spaces: 4 UTF-8 CRLF Python

Another language!

Let's ***not only*** add another language...

... ***but also make it half the hw!***



*Even with three eyes, I
must be misreading this!*

Another language *already*?

Python

General-purpose language

you might see
50% by the end
of the term

even then, <1% of its libraries!

Picobot

Special-purpose language

you'll see 100% in
the next 10 minutes

Picobot!

Picobot

Rules

```
# These lines are comments.  
# Remember that rules are formatted as  
# State Surroundings -> Move NewState  
  
# Picobot starts in state 0.  
# Here, state 0 goes N as far as possible  
0 x*** -> N 0 # if there's nothing to the N, go N  
0 N*** -> X 1 # if N is blocked, switch to state 1  
  
# and state 1 goes S as far as possible  
1 ***x -> S 1 # if there's nothing to the S, go S  
1 ***0 -> X 0 # otherwise, switch to state 0
```

Enter rules for Picobot

Be sure to hit "Enter rules" after making changes.

Messages

OK

Go Stop Step Reset <-- MAP -->

0 State xxxxx Surroundings 528 Cells to go

Previous Rule Next Rule

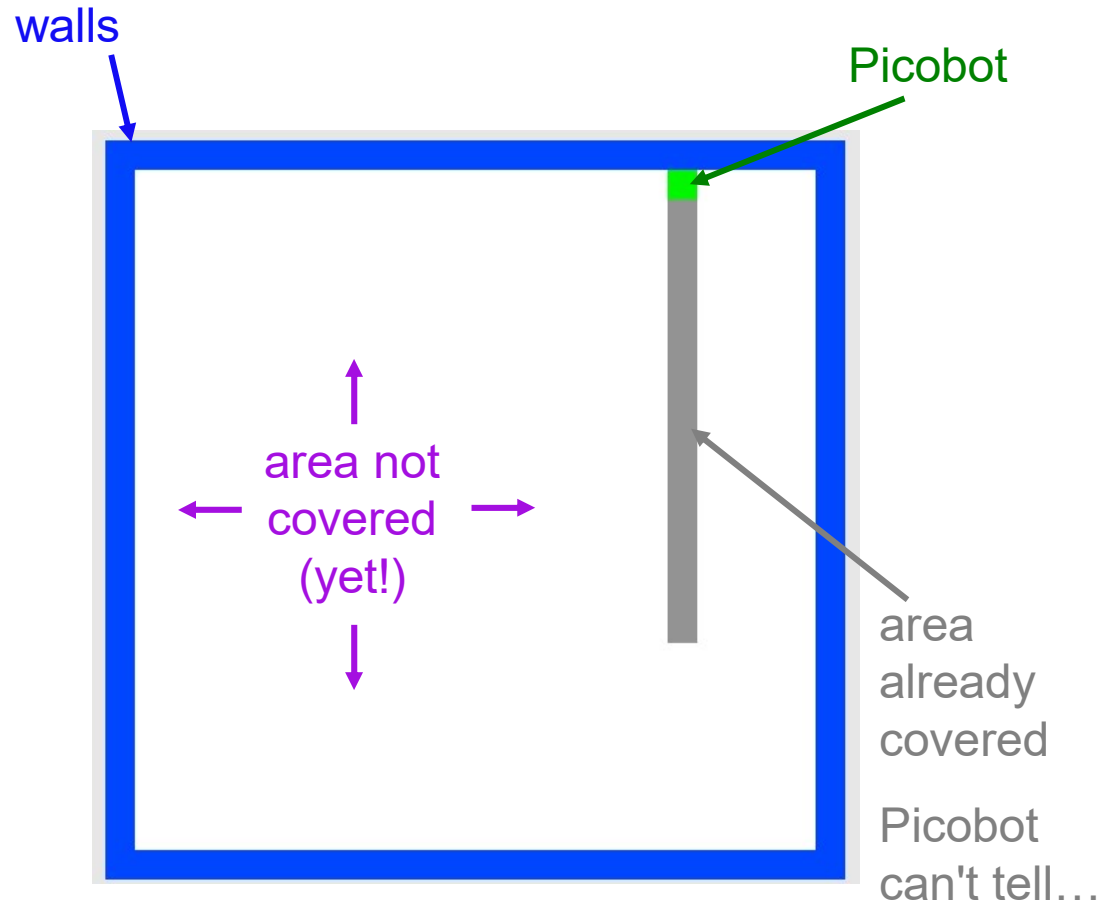
West East - Teleport Robot - North South

The Picobot simulator
www.cs.hmc.edu/picobot

Demo!

HW problems 3 and 4: Picobot!

Goal: full-room coverage with only *local sensing*...



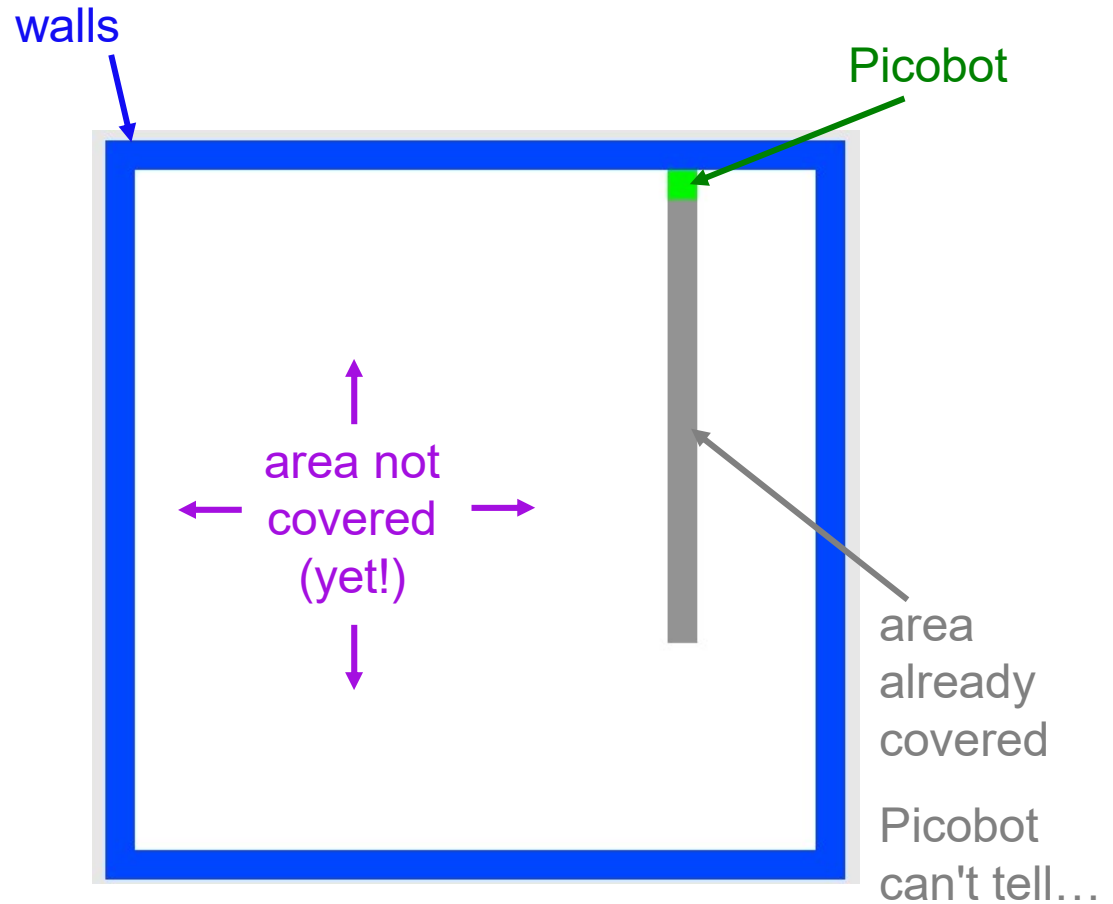
Inspiration?

HW problems 3 and 4: Picobot!

Goal: full-room coverage with only *local sensing*...

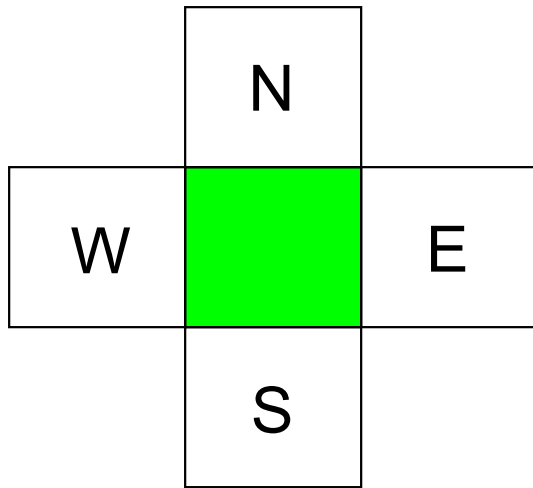


The Roomba!
can't tell "vacuumed"
from "unvacuumed" area



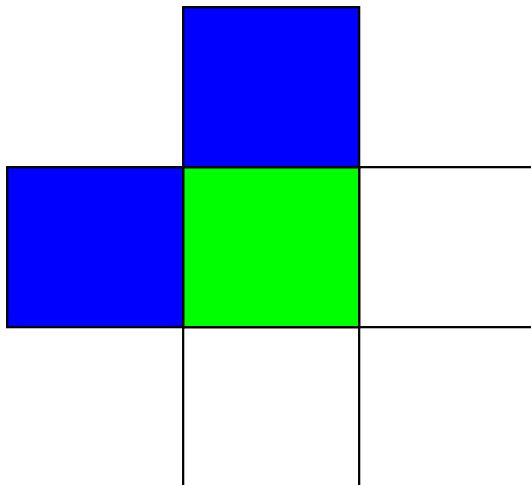
Let's see it!

Surroundings



Picobot can only sense things directly to the N, E, W, and S

For example, here its surroundings are



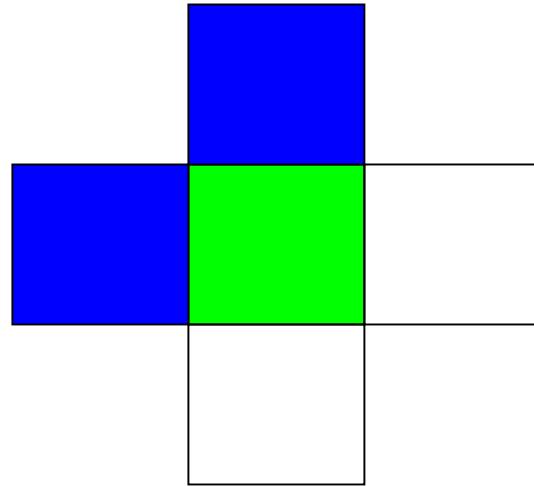
NxWx

N E W S

Surroundings are always in **NEWS** order.

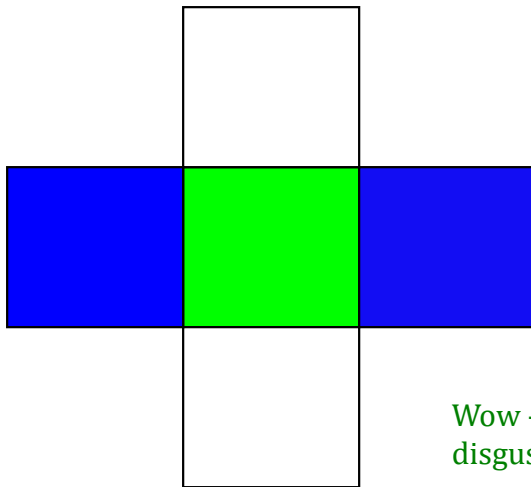
What are these surroundings?

Surroundings are always in **NEWS** order.

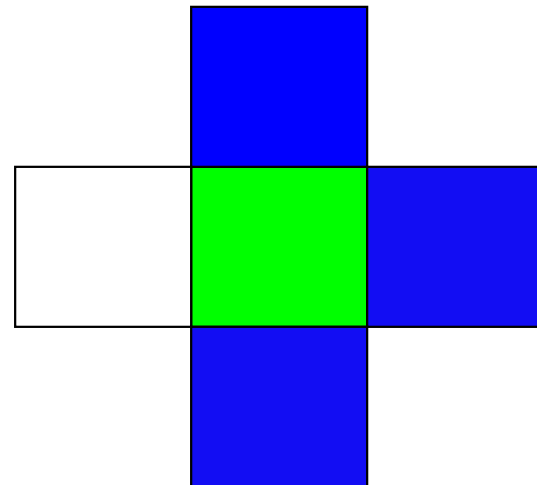


N E W S

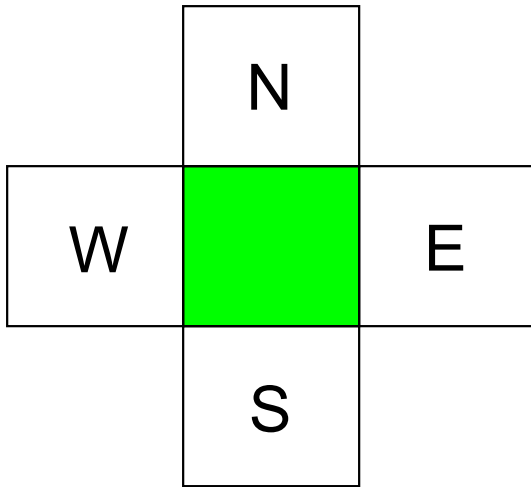
NxWx



Wow - this one is disgusting!

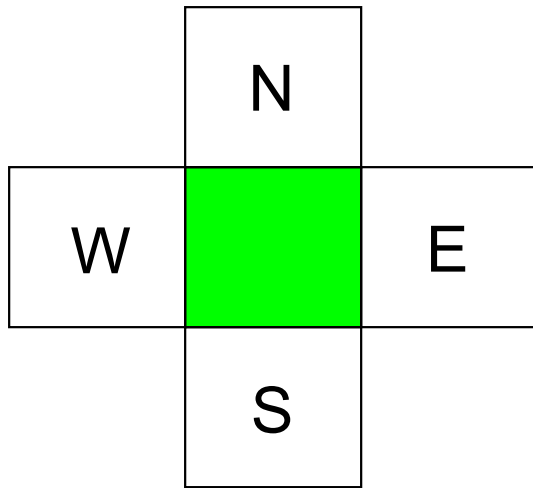


Surroundings



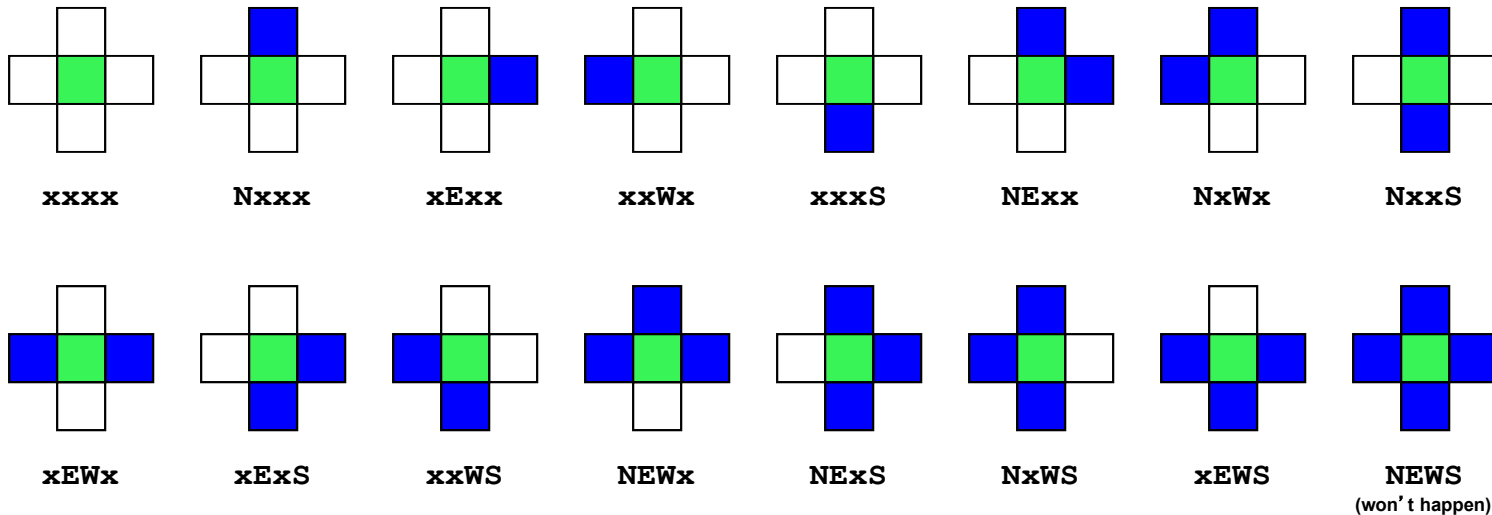
How many distinct
surroundings are there?

Surroundings



How many distinct surroundings are there?

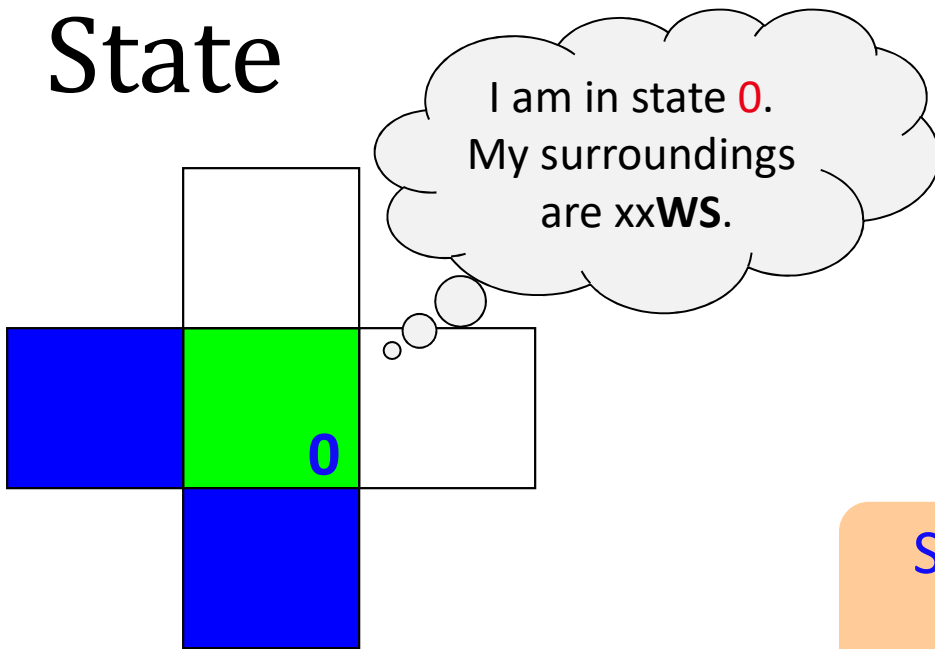
$2^4 == 16$ possible



Aargh!



State



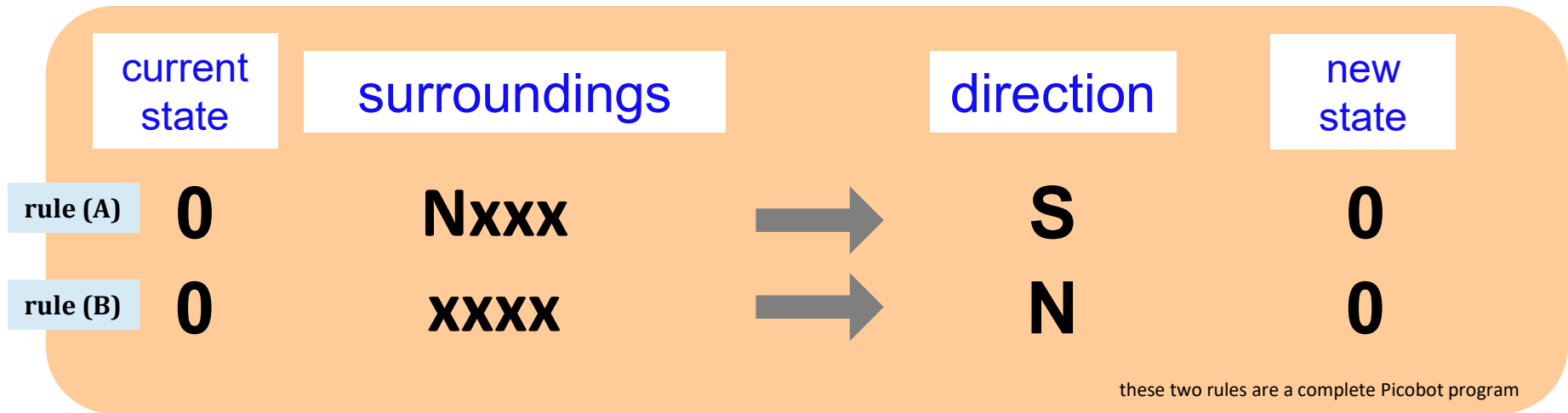
Picobot's memory is a single number, called its **state**.

State is the *internal context* of a computation, i.e., its *subtask*.

Picobot always starts in **state 0**.

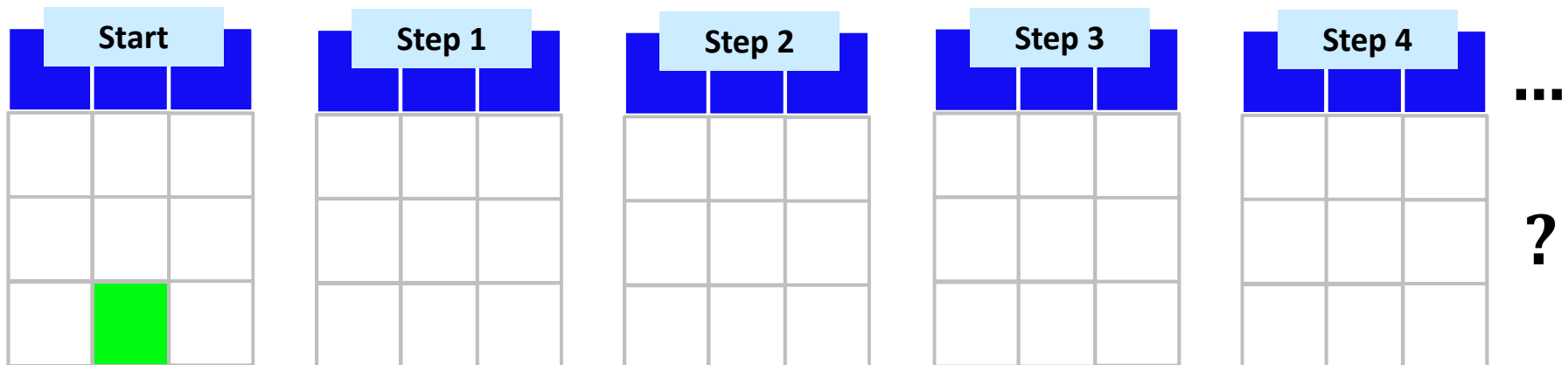
State and **surroundings** represent everything Picobot knows about the world

Picobot programming ~ *rules*

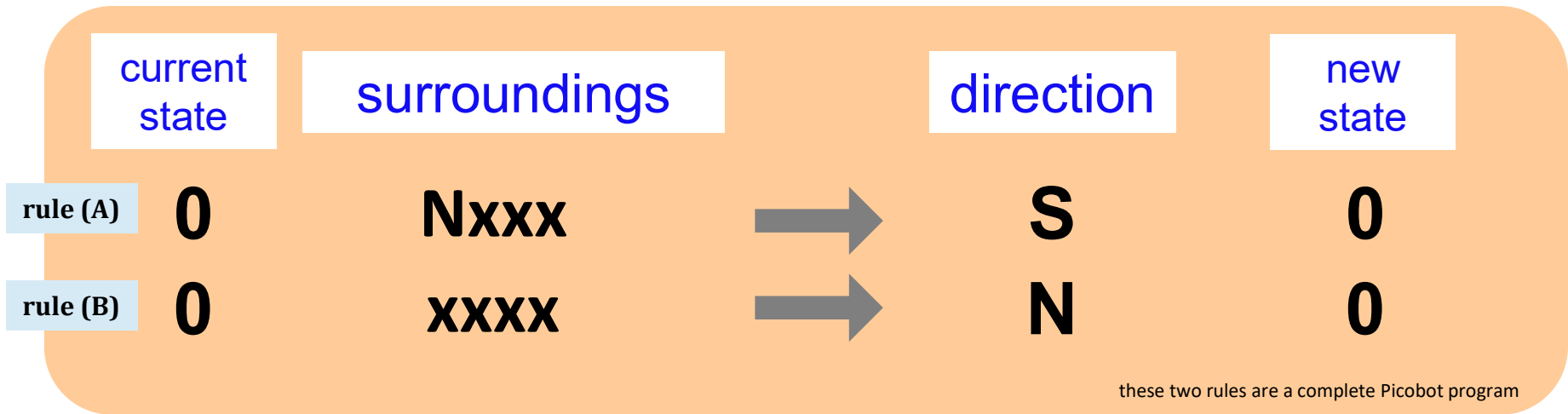


Notes

Picobot checks its rules from the top each time.
When it finds a matching rule, that rule runs.

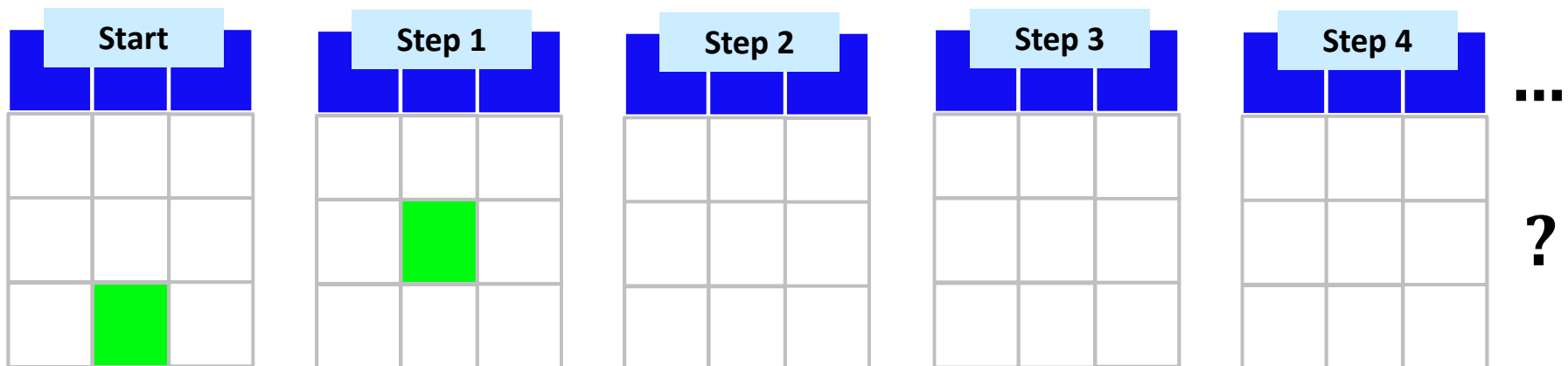


Picobot programming ~ *rules*

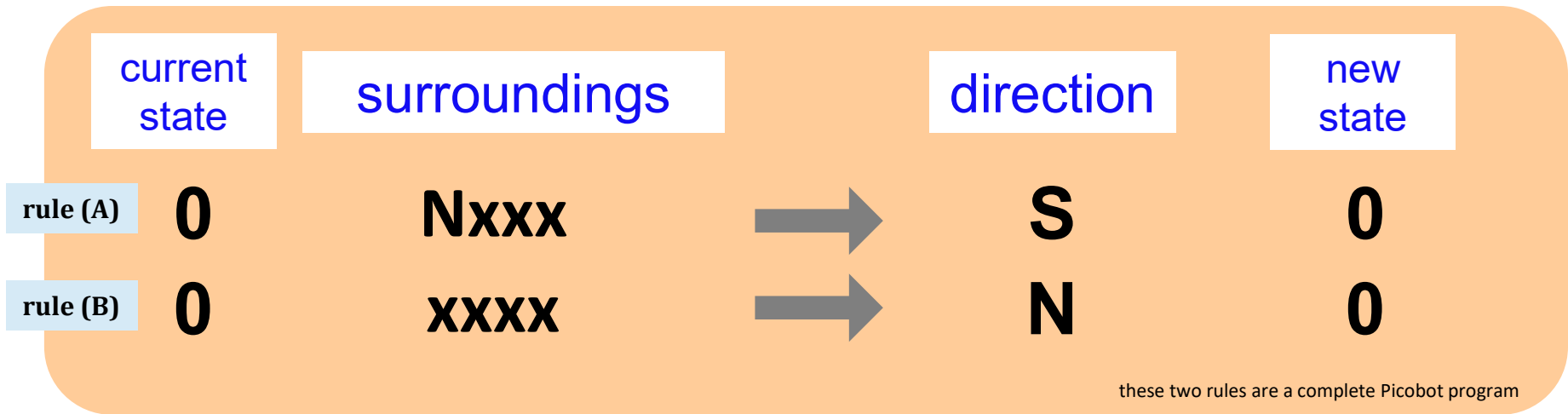


Notes

Picobot checks its rules from the top each time.
When it finds a matching rule, that rule runs.

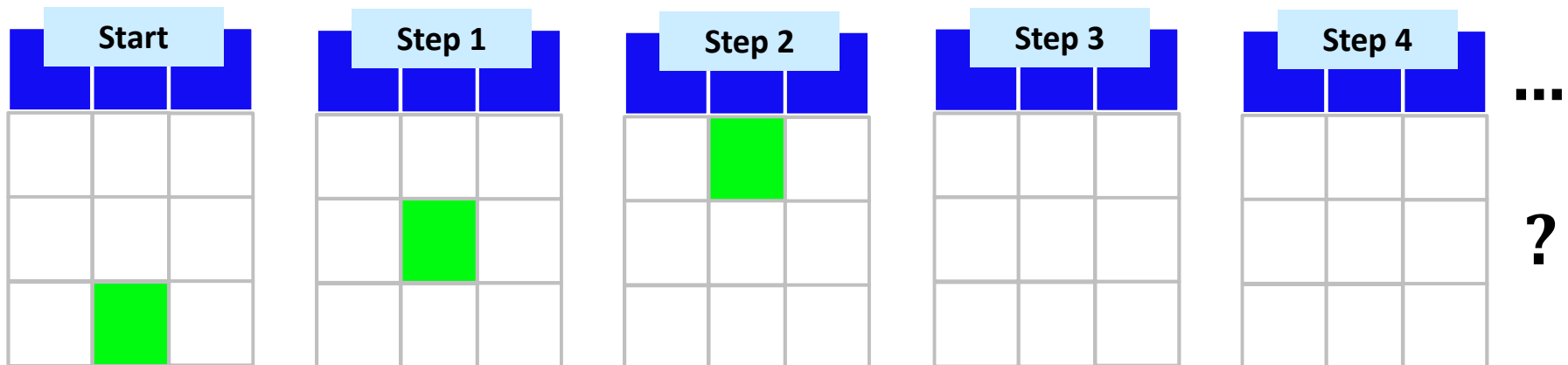


Picobot programming ~ *rules*

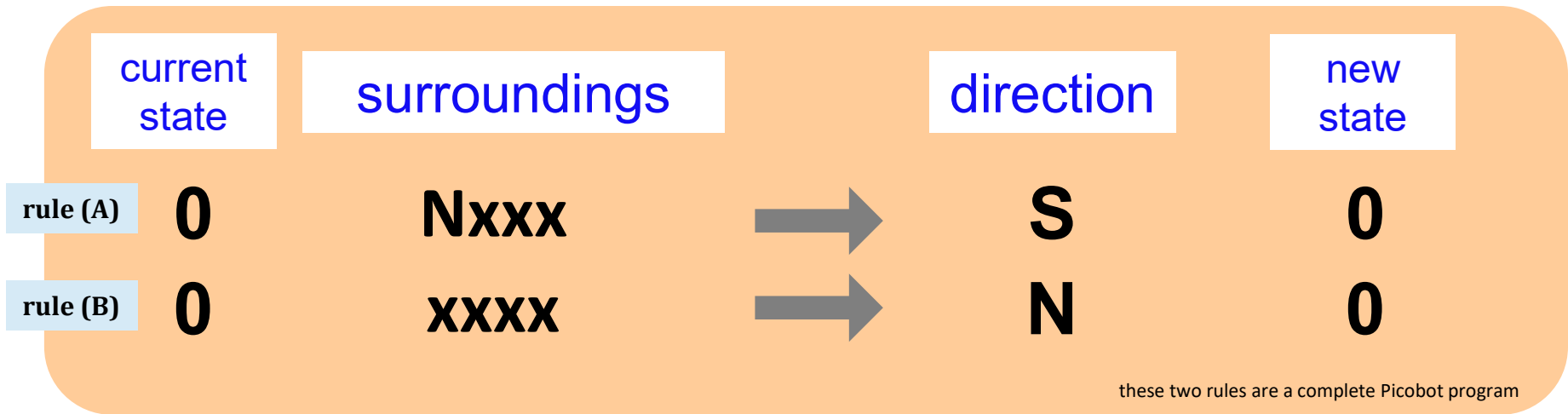


Notes

Picobot checks its rules from the top each time.
When it finds a matching rule, that rule runs.

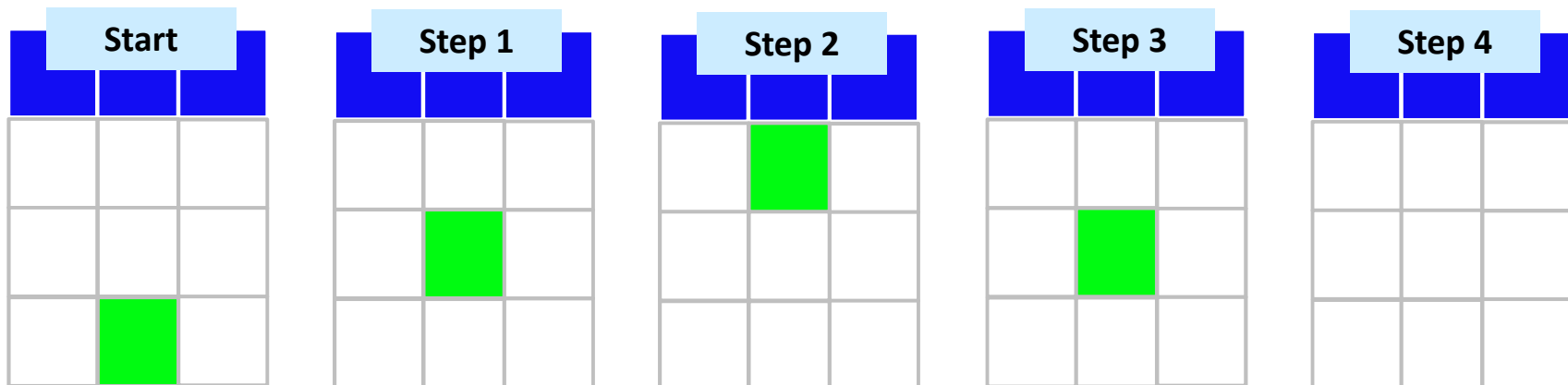


Picobot programming ~ *rules*

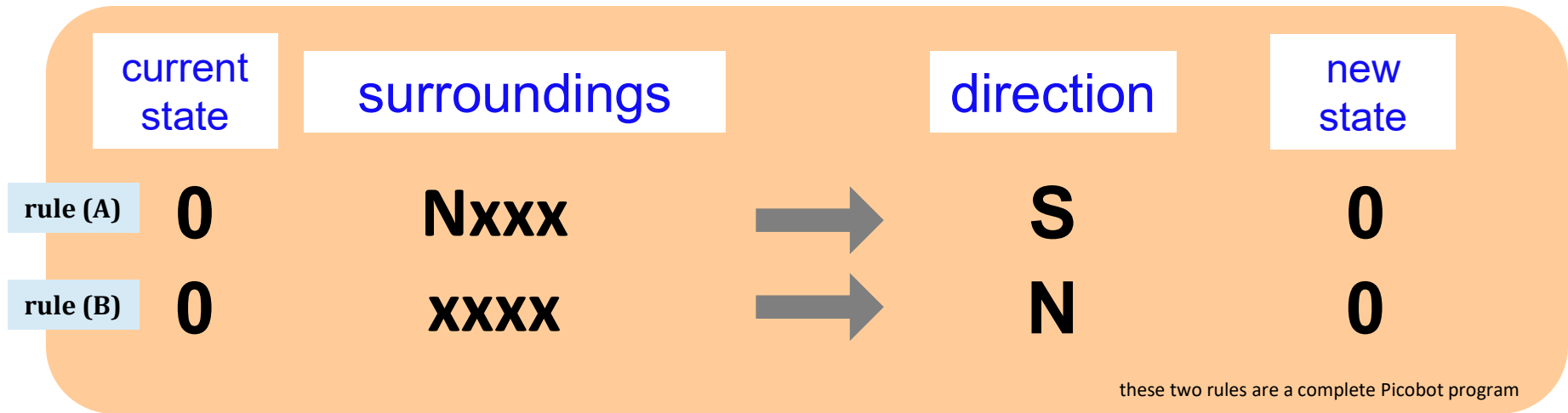


Notes

Picobot checks its rules from the top each time.
When it finds a matching rule, that rule runs.

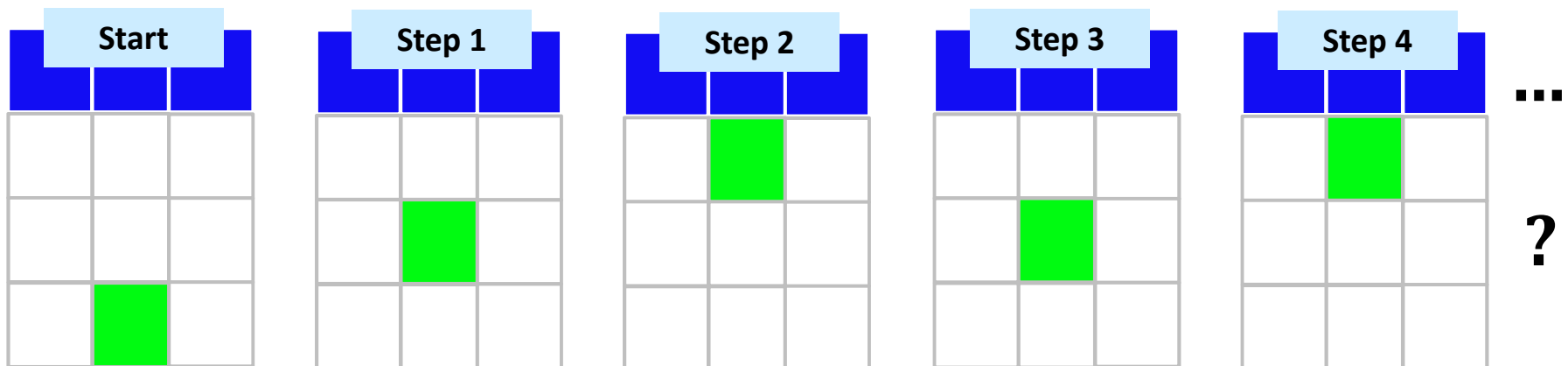


Picobot programming ~ *rules*

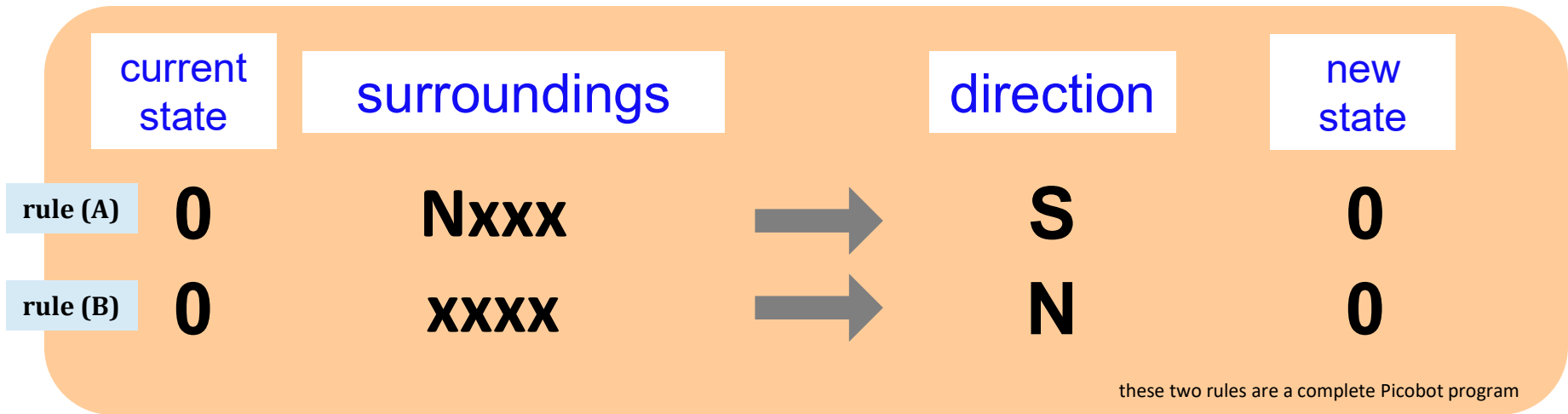


Notes

Picobot checks its rules from the top each time.
When it finds a matching rule, that rule runs.

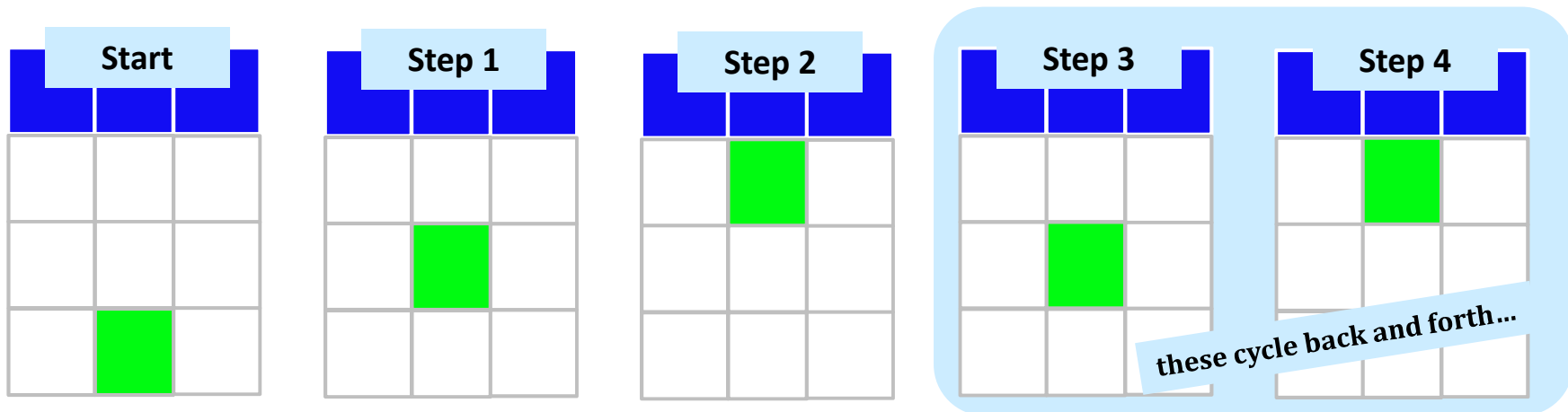


Picobot programming ~ *rules*

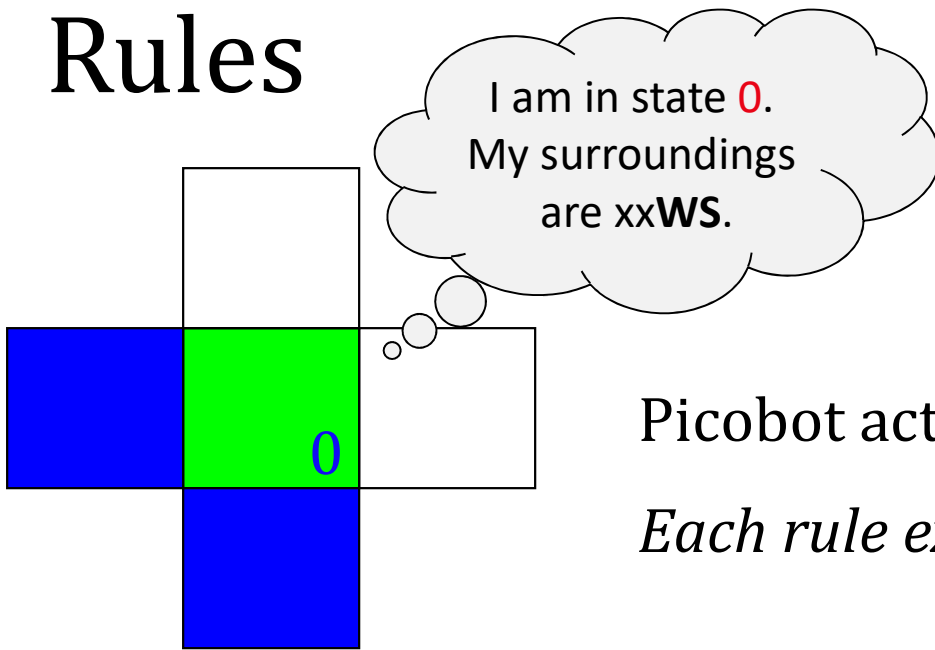


Notes

Picobot checks its rules from the top each time.
When it finds a matching rule, that rule runs.

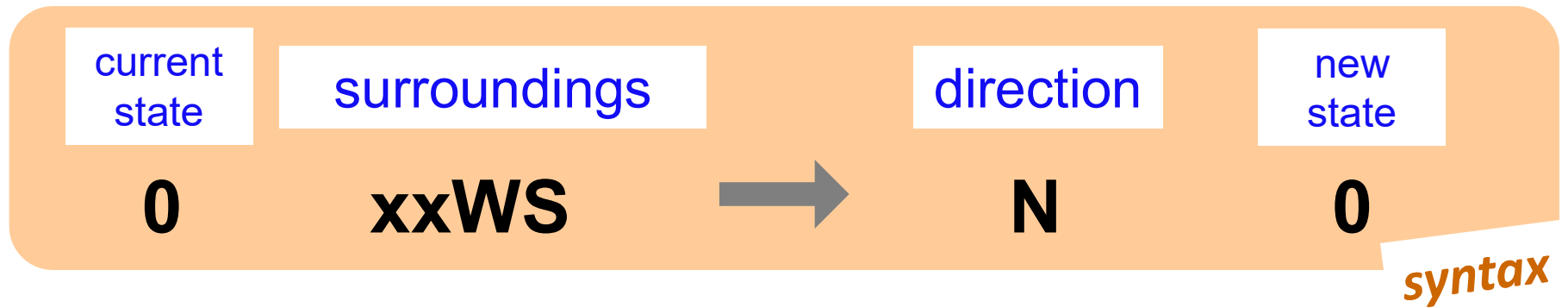


Rules



Picobot acts through a set of rules

*Each rule expresses **your intent** for Picobot!*



*If Picobot's in state
0 seeing **xxWS**,*

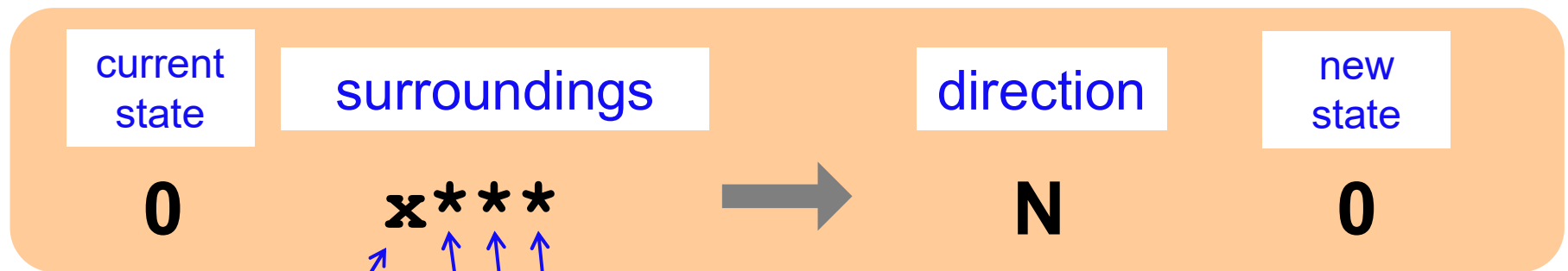
*Then move **N**orth, and
"change" to state **0**.*

semantics

Wildcards

I only care about **NORTH** being **EMPTY**

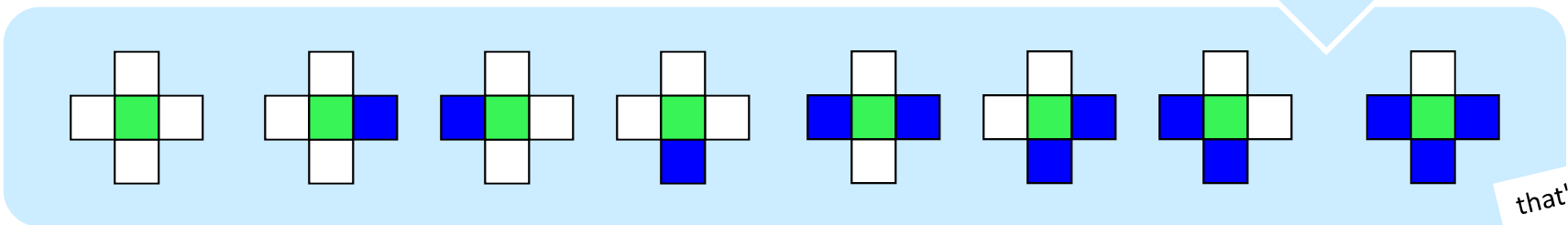
Asterisks * are wild cards.
They match walls *or* empty space:



N must be empty

EWS may be wall *or* empty space

8 surroundings
in one rule



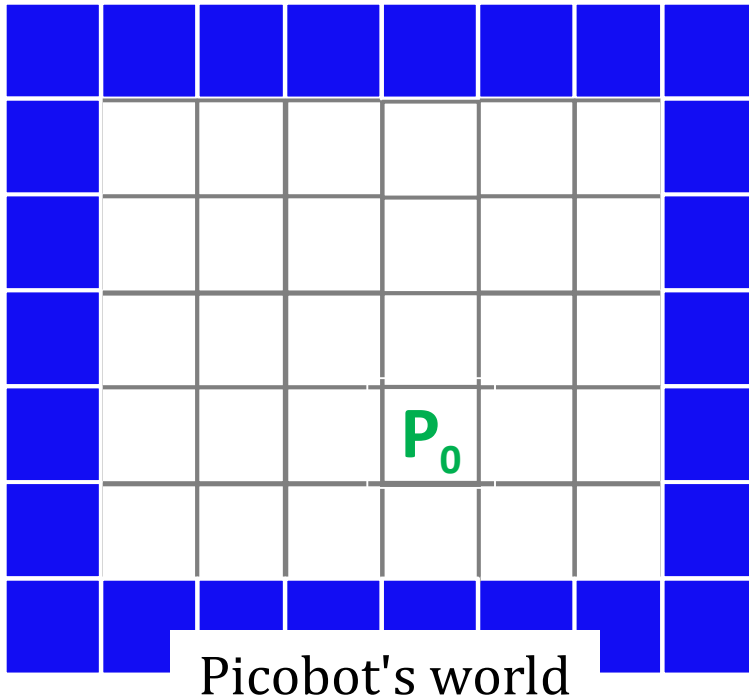
that's it!

The Rule is *One step per rule*

One rule to rule them all?



That's *precious!*



	state	surr.	move	new state
rule (A)	0	N***	-> W	1
rule (B)	0	x***	-> N	0
rule (C)	1	***x	-> S	1
more rules				

1. Run Picobot! Which rule **A**, **B**, or **C** runs *first*? _____
 - 1a. How many times does **rule (A)** run? _____
 - 1b. How many times does **rule (B)** run? _____
 - 1c. How many times does **rule (C)** run? _____

2. Picobot stops when no rule matches. *Where does it stop?*
3. Add a rule so that Picobot continues *back upward!*

Extra #1 Rule A has a bug! What is it?
Extra #2 Add rules to finish exploring the empty room *from any starting point...*
Extra #3 *How to do this in only 6 rules total?!*

Warning! *What's wrong here?*

state	surroundings		direction	new state
0	x***	→	S	0
0	***x	→	N	0

these two rules are a broken Picobot program!

Notes

Picobot checks its rules from the top each time.
When it finds a matching rule, that rule runs.

Warning! *What's wrong here?*

state	surroundings	direction	new state
0	x***		
0	***x		

These two situations COULD BE the same!

these two rules are a broken Picobot program!

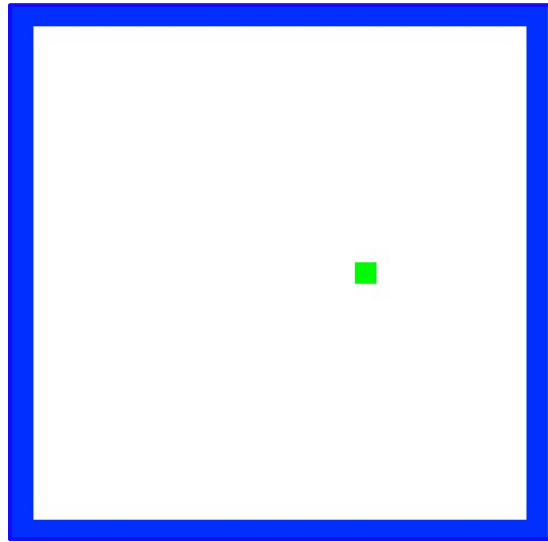
Notes

Picobot checks its rules from the top each time.
When it finds a matching rule, that rule runs.

There can only be **ONE** rule per situation!

and a "situation" is *state* and *surroundings*

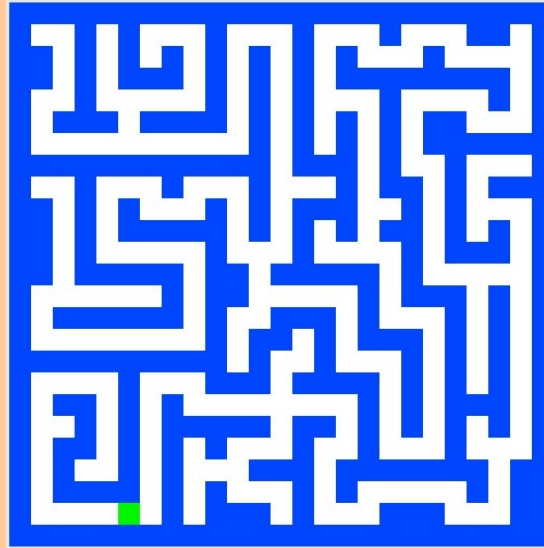
CS ~ Complexity Science



problem 3

Shortest Picobot
program:

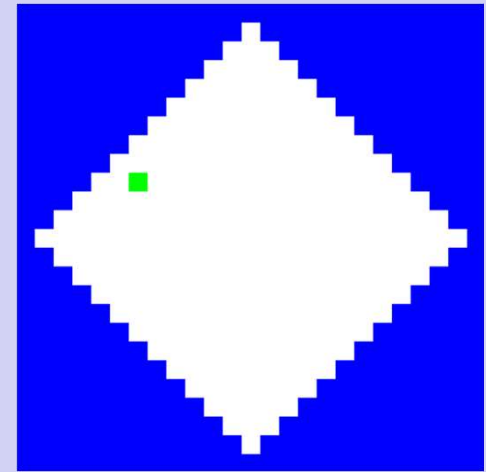
6 rules



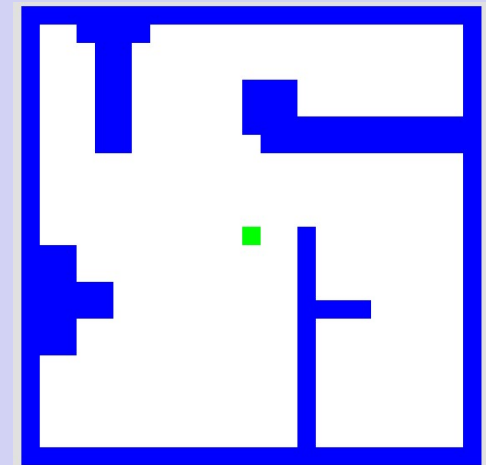
problem 4

Shortest Picobot
program:

8 rules

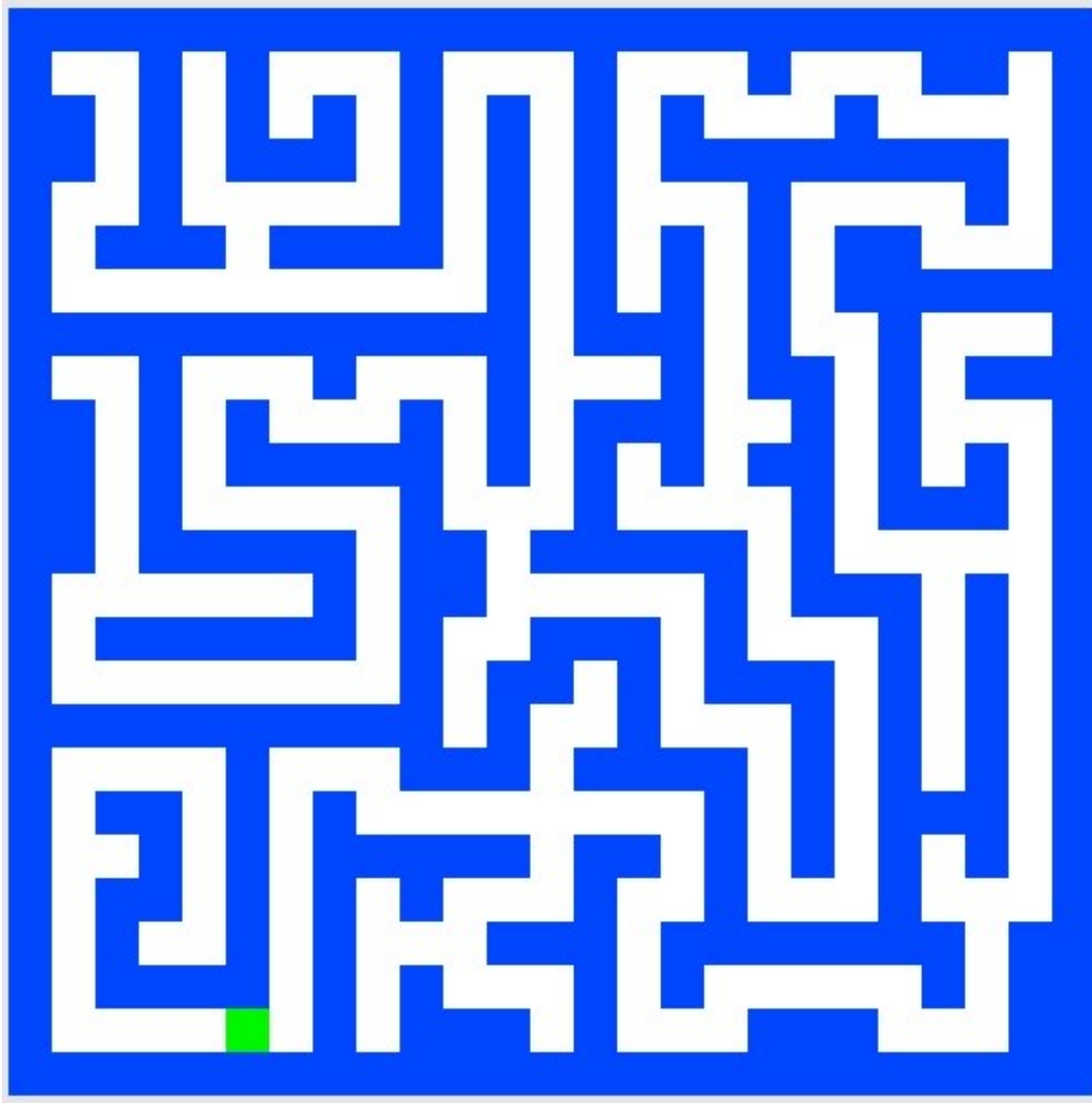


pr. 5 (extra!)



pr. 6 (extra!)

Maze strategies?





WIKIPEDIA
The Free Encyclopedia

- Main page
- Contents
- Featured content
- Current events
- Random article
- Donate to Wikipedia
- Wikimedia Shop

- Interaction
 - Help
 - About Wikipedia
 - Community portal
 - Recent changes
 - Contact page

Article

Talk

Read

Edit

View history

Search

Maze solving algorithm

From Wikipedia, the free encyclopedia

There are a number of different **maze solving algorithms**, that is, automated methods for the solving of **mazes**. The random mouse, wall follower, Pledge, and Trémaux **algorithms** are designed to be used inside the maze by a traveler with no prior knowledge of the maze, whereas the **dead-end** filling and **shortest path algorithms** are designed to be used by a person or computer program that can see the whole maze at once.



Mazes containing no loops are known as "standard", or "perfect" mazes, and are equivalent to a *tree* in graph theory. Thus many maze solving algorithms are closely related to **graph theory**. Intuitively, if one pulled and stretched out the paths in the maze in the proper way, the result could be made to resemble a tree.^[1]

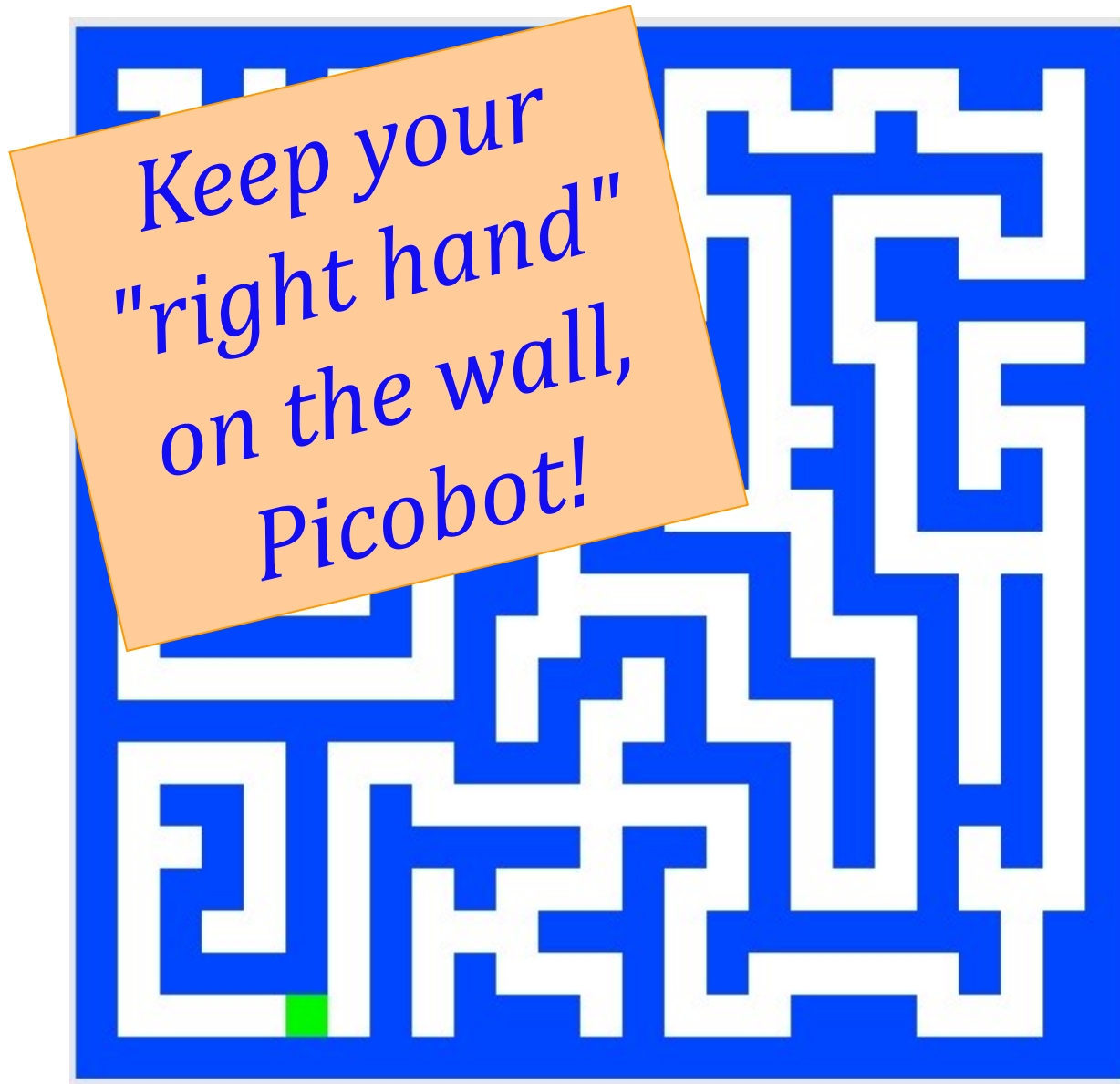
Contents [hide]

- 1 Random mouse algorithm

Right Hand Rule

Maze strategies?

Right Hand Rule



Why might this be *difficult* for Picobot?

Maze strategies?

Right Hand Rule

Keep your
"right hand"
on the wall,
Picobot!



facing

to the
right

State **0**

State **1**

State **2**

State **3**

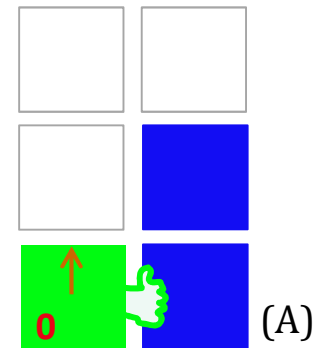
We'll need to use
state to represent
the *direction*
Picobot is facing.

Suppose Picobot wants to traverse a maze *with its right hand always on the wall...*

(A) CORRIDOR rule

“If you're facing N with a wall at right and space ahead then go forward”

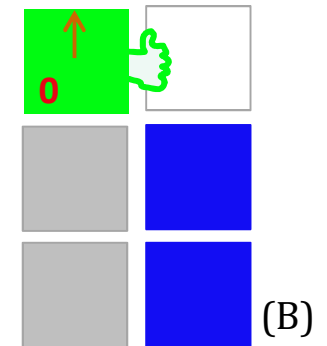
0 **xE**** -> N 0



(B) INTERSECTION rule

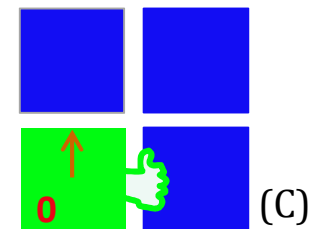
“If you're facing North and lose the wall, then get over to the wall now!”

0 ->



(C) DEAD END rule

Write 1 or 2 rules to tell Picobot to do the right thing if it hits a dead end.



Repeat this IDEA for all four states, representing all four *facing directions*.

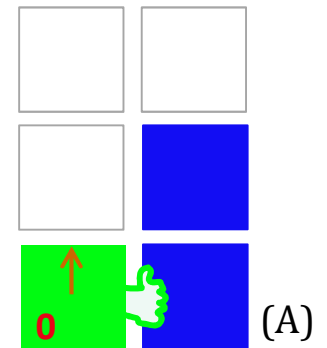
Suppose Picobot wants to traverse a maze *with its right hand always on the wall...*

(A) CORRIDOR rule

"If you're facing N with a wall at right and space ahead then go forward"

0 **xE**** -> N 0

state 0 means "still facing north"

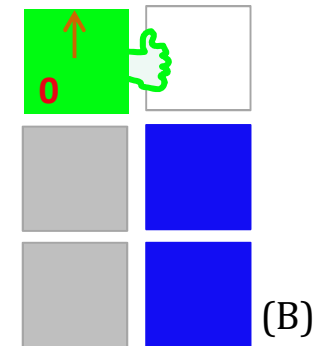


(B) INTERSECTION rule

"If you're facing North and lose the wall, then get over to the wall now!"

0 ***x**** -> E 1

state 1 means "now facing east"

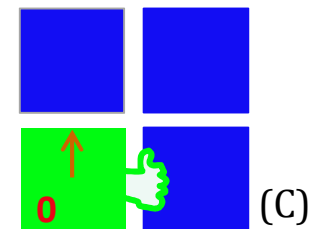


(C) DEAD END rule

Write 1 or 2 rules to tell Picobot to do the right thing if it hits a dead end.

0 **NE**** -> X 2

state 2 means "now facing west"



Repeat this IDEA for all four states, representing all four *facing directions*.

Hooray!?!

Picobot

Rules

Is it working?

```
## twelve-rule maze-solver:
```

Enter rules for Picobot

Be sure to hit "Enter rules" after making

Messages

Go Stop Step Reset <-- MAP -->

Southward!

Northward!

- *Onward* -

Westward!

Eastward!

Lab/hw

You are not alone!

Come to tutoring hours!

Post questions to piazza...

I can attest
to that!



Happy Picobotting!

Lead on!
I will follow.



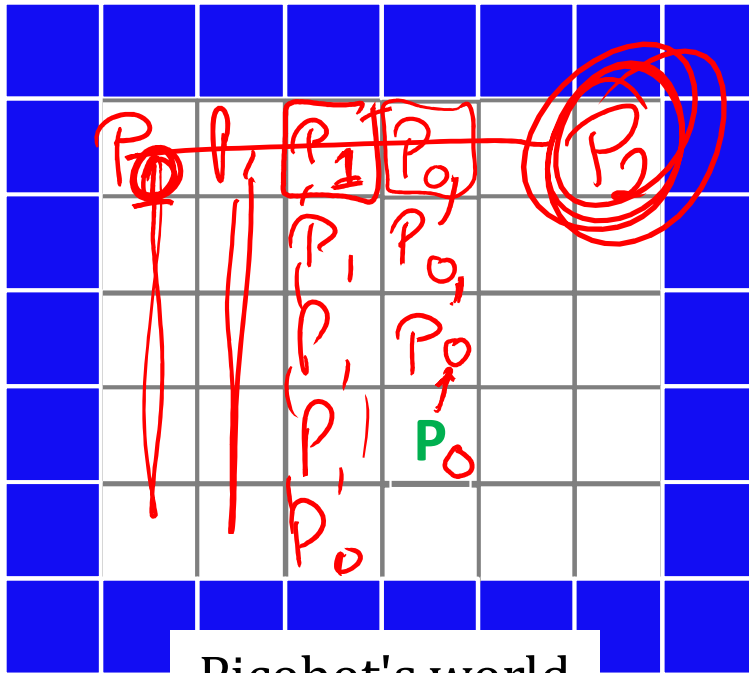
*And, good luck with the **adventure** of Python!*

The Rule is *One step per rule*

One rule to rule them all?



That's precious!



Picobot's world

keep going up

	state	surr.	move	new state
rule (A)	0	N***	-> W	1
rule (B)	0	x***	-> N	0
rule (C)	1	***x	-> S	1
more rules	1	***S	-> X	0
	0	N**	-> E	2

keep S

- Run Picobot! Which rule **A**, **B**, or **C** runs *first*? B
 - How many times does **rule (A)** run? 1
 - How many times does **rule (B)** run? 111
 - How many times does **rule (C)** run? 111

- Picobot stops when no rule matches. *Where does it stop?*
- Add a rule so that Picobot continues *back upward!* \square

Extra #1 Rule A has a bug! What is it?
 Extra #2 Add rules to finish exploring the empty room from any starting point...
 Extra #3 How to do this in only 6 rules total?!

Try this on the back page first... !

(1) Find and correct as many errors as you can in this code:

Syntax challenge!

```
import random
```

(2) This one line does *three* things... what are they?



```
user = input( "Choose your weapon! " )
```

```
comp = random.choice( ['rock', 'paper', 'scissors'] )
```

```
print('user (you) chose:', 'user')
```

```
print('comp (me!) chose:' comp)
```

```
if user == rock and comp = 'paper'
```

```
    print('The result is, YOU LOSE.'
```

```
    print('unless you're a CS 5 grader, then YOU WIN!')
```

(3) *Extra!* Can you find 7 punctuation marks used in *more than one way* here?

By the Koi pond!



A different view...



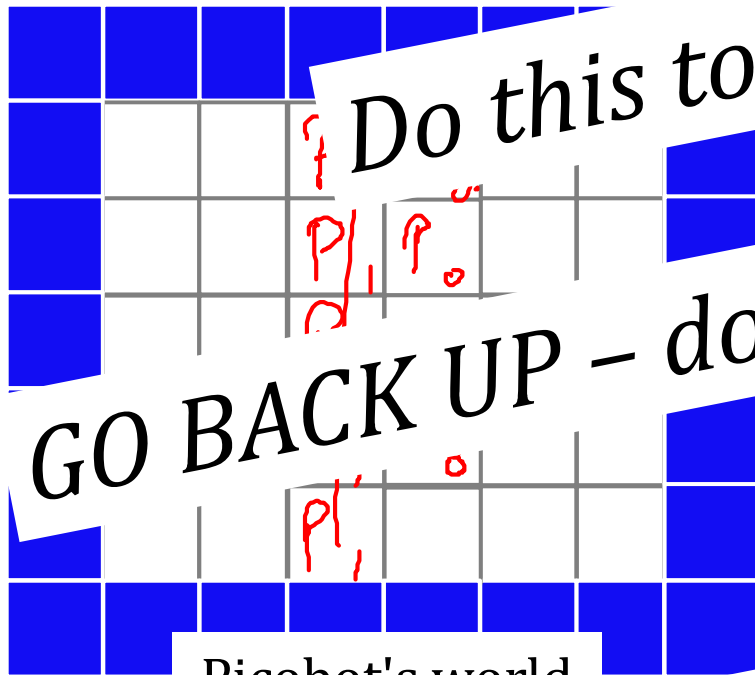
Try it!

The Rule is One step per rule

One rule to rule them all?



That's precious!



Picobot's world

Do this together!

GO BACK UP - do not go

state	surr.	new state
0	x***	N 0
1	***x	S 1

rule (B)

rule (C)

W on 1!!!!!!

Ask how to extend + do it
 - if get to the corner first, great!
 - if get back across, great!

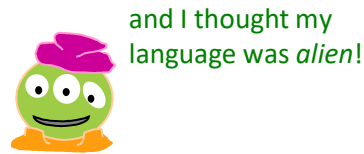
1. Which of
2. Picobot st

Once Picobot stops,

3. - if get to the corner first, great!
4. - if get back across, great!
5. rule (C) run? _____

6. Add a 4th rule so that Picobot can continue westward.

Extra! Add rules to finish exploring the empty room...
 Can you do it in 6 rules total?



Does Your Language Shape How You Think?



Seventy years ago, in 1940, a popular science magazine published a short article that set in motion one of the trendiest intellectual fads of the 20th century. At first glance, there seemed little about the article to augur its subsequent celebrity. Neither the title, “Science and Linguistics,” nor the magazine, M.I.T.’s Technology Review, was most people’s idea of glamour. And the author, a chemical engineer who worked for an insurance company and moonlighted as an anthropology lecturer at Yale University, was an unlikely candidate for international superstardom. And yet Benjamin Lee Whorf let loose an alluring idea about language’s power over the mind, and his stirring prose seduced a whole generation into believing that our mother tongue restricts what we are able to think.

Seventy years ago, in 1940, a popular

But then a remote Australian aboriginal tongue, Guugu Yimithirr, from north Queensland, turned up, and with it came the astounding realization that not all languages conform to what we have always taken as simply “natural.” In fact, Guugu Yimithirr doesn’t make any use of egocentric coordinates at all. The anthropologist John Haviland and later the linguist Stephen Levinson have shown that Guugu Yimithirr does not use words like “left” or “right,” “in front of” or “behind,” to describe the position of objects. Whenever we would use the egocentric system, the Guugu Yimithirr rely on cardinal directions. If they want you to move over on the car seat to make room, they’ll say “move a bit to the east.” To tell you where exactly they left something in your house, they’ll say, “I left it on the southern edge of the western table.” Or they would warn you to “look out for that big ant just north of your foot.” Even when shown a film on television, they gave descriptions of it based on the orientation of the screen. If the television was facing north, and a man on the screen was approaching, they said that he was “coming northward.”



believing that our mother tongue restricts what we are able to think.

- *Southward, Ahoy!* -

Lab/hw

You are not alone!

Come to tutoring hours!

Post questions to piazza...

I can attest
to that!



Happy Picobotting!

Lead on!
I will follow.



*And, good luck with the **adventure** of Python!*