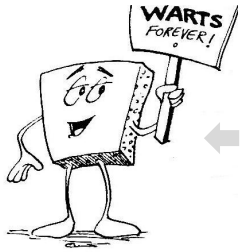
Welcome back to CS 5 !



Wally

Average of these two?

Alien

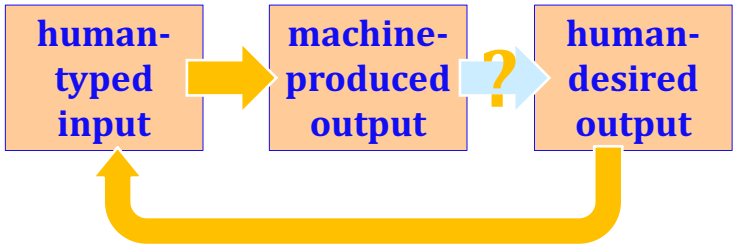
Read Section 1.2

Homework 1
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)

The *challenge* of programming...

syntax	semantics	intent
How it looks	What it does	What it should do



Another language *already*?

Python

General-purpose language

you might see 50% by the end of the term

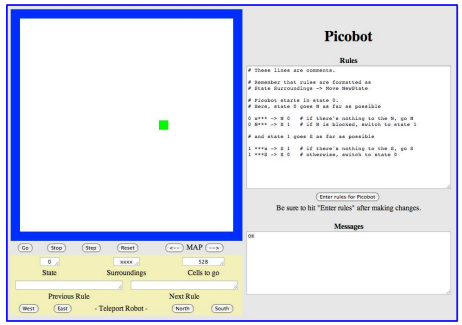
even then, <1% of its libraries!

Picobot

Special-purpose language

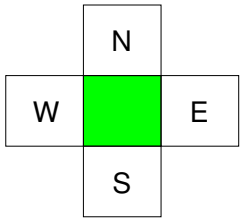
you'll see 100% in the next 10 minutes

Picobot!



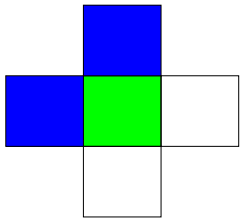
The Picobot simulator
www.cs.hmc.edu/picobot

Surroundings



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

For example, here its surroundings are



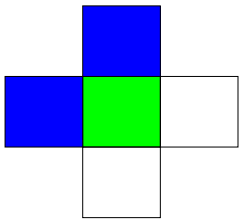
NxWx

N E W S

Surroundings are always in NEWS order.

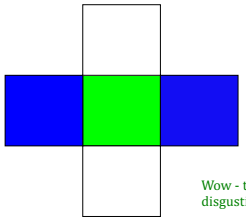
What are these surroundings?

Surroundings are always in NEWS order.

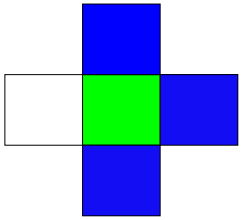


N E W S

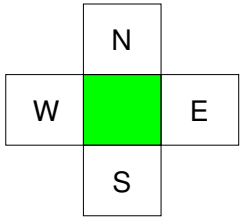
NxWx



Wow - this one is disgusting!

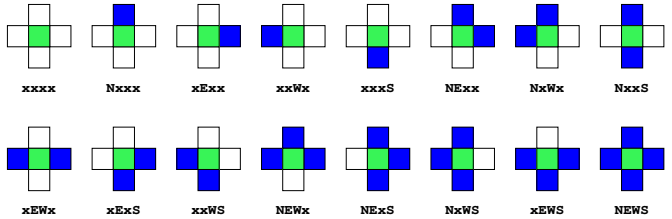


Surroundings



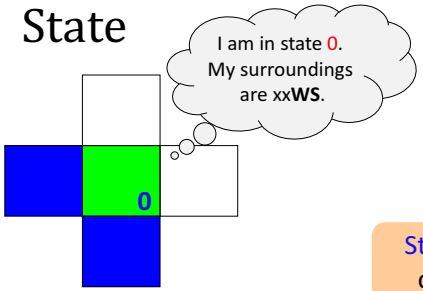
How many distinct surroundings are there?

2^4 == 16 possible



Aargh!

State



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

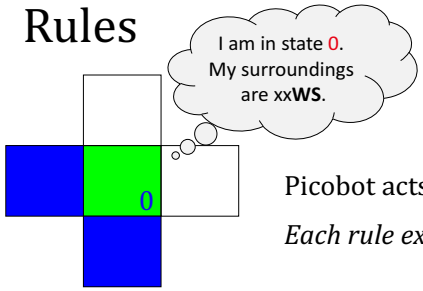
State is the *internal context* of a computation, i.e., its *subtask*.

Picobot always starts in state 0.

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

self-contained but not simplistic

Rules



Picobot acts through a **set of rules**

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

current state	surroundings		direction	new state
0	**xxWS**	→	**N**	**0**

syntax

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

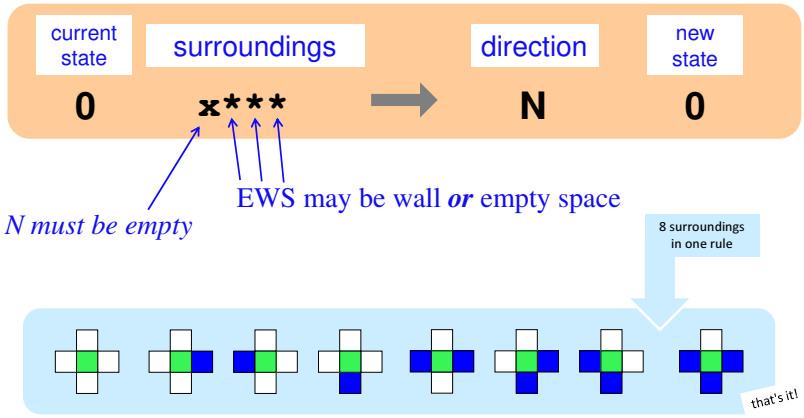
Then move **N***orth, and "change" to state* **0**.

semantics

Wildcards

I only care about **NORTH being EMPTY**

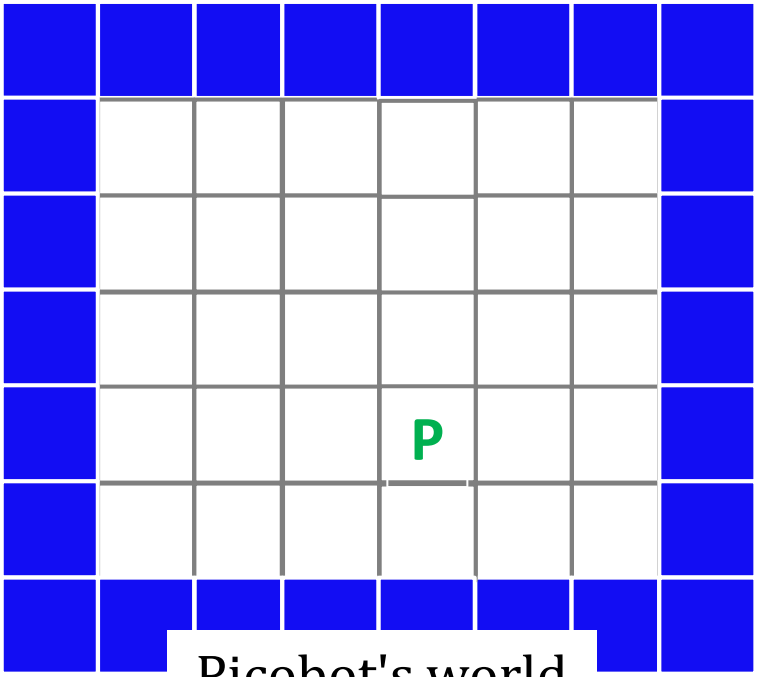
Asterisks * are wild cards.
They match walls ***or*** empty space:



The Rule is ***One step per rule***

One rule to rule them all?

That's *precious*!



Picobot's world

	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

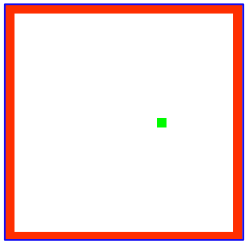
these two rules are a broken Picobot program!

Notes

Picobot checks its rules from the top each time.

When it finds a matching rule, that rule runs.

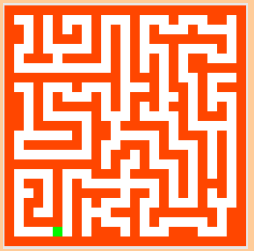
CS ~ *Complexity* Science



problem 3

Shortest Picobot program:

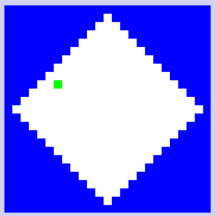
6 rules



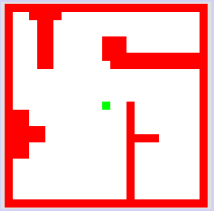
problem 4

Shortest Picobot program:

8 rules

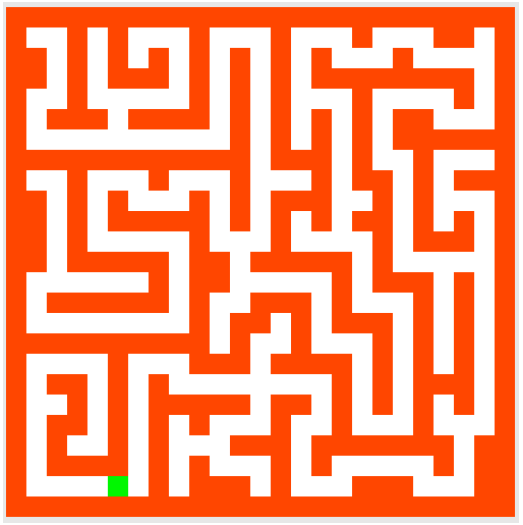


pr. 5 (extra!)



pr. 6 (extra!)

Maze strategies?



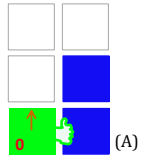
Suppose Picobot wants to traverse a maze ***with its right hand always on the wall...***

(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"

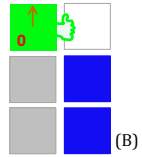


(A)

(B) INTERSECTION rule

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

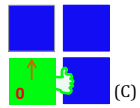
0	->	



(B)

(C) DEAD END rule

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



(C)

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

Name(s):

Syntax challenge!

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

```
import random

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!')
```

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

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