Welcome back, CS!

Finally, I'm concrete!
To begin to visualize the city’s government of its familiarity. In The
implications of omitting a small town like Claremont, California may believe that the
general purpose of the city and its familiarity.

So don't be there. Just call me up 'Cause I see your true colors. Just call me a rainbow Ooh oh oh oh like a rainbow Oh oh oh oh oh
oh This world makes you crazy. Just call me a world makes you Can make you feel so small.

"Is it was advancing on for Harry a broomstick, Professor," said Harry, fighting not any questions?" Harry gasped. Everyone stared as it with their way upstairs with the yell and said, "it's a Nimbus Two Thousand turned wherever he prodded it when the Bloody Baron. Hermione snap.
"It's either end of them look up, may go."
CS5's view from here...

What's next?

<table>
<thead>
<tr>
<th>Week 13</th>
<th>4/23/19: Finite-state machines (Section 1) (Section 2)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>4/25/19: Turing machines! (Section 1) (Section 2)</td>
</tr>
<tr>
<td>Week 14</td>
<td>4/30/19: Review-via-debugging and Undebuggable functions! (Section 1) (Section 2)</td>
</tr>
<tr>
<td></td>
<td>5/2/19: Finale (Section 1) (Section 2)</td>
</tr>
<tr>
<td>Finals Week</td>
<td>5/14/19: Final Exam 2-5pm or 7-10pm Final Review</td>
</tr>
</tbody>
</table>

... and the CS5 finale.

**CS 5 final exam review materials**

The final exam for CS 5 takes place on Tuesday, May 14th, 2019 at 2pm (Section 1) or 7pm (Section 2)

- The review session will be Thursday, May 9th, 2019 at our usual class time (8:10am or 9:35am).

Both are in Shan B442, our usual room.

There will be nearby rooms for extended/distraction-minimized exam-taking.
vPython's *key-to-all-happiness*

Make **THE MOST BORING POSSIBLE** version of your game...

*everything's a sphere... (maybe different colors...)*

but, only *at first*...

*... LATER, add more intricate models.*
Any other approach & you may "lose your mind..."!

OR ... at least develop in two different files!

HP3: Prisoner of Azkaban
Final project: milestone/final

Due 4/29: Final project "milestone" (6-7 functions)

Due Fri., 5/3 (8pm): Final project (final version)
Labs!

are *optional* meeting times:

| Labs:       | Tuesdays          | 1:10-3:10 pm and 3:15-5:15 pm |

- to work on *projects*: start/milestone/final
- to work on hw 12's *finite-state machines*

We *won't* be able to get you graded feedback on the milestone before the final project is due – *so join us for lab!*
Homework 12

(1) **project milestone** ~ "progress report"

For the milestone, you’ll submit a new `.py` file:

- `milestone.py`
  - worth 25 points

(2) **Building** **finite-state machines**

50 assigned points; **up to +30 ex. cr. pts; due by 4/29**
CS5's broad view:

**CS Applications**
- simulation + analysis/algorithms
- graphics / media / games ...
- feature-based modeling + classification
- other state-machines + fun stuff

**CS 5**
- recursion
- variables
- loops
- functions
- circuits and memory
- data: classes and objects

**CS Theory**
- What *can* we compute...
- ... and *how well*?

**Final Projects**
- Picobot
- vPool
- TextID

**CS Practice**

What's next?

is under here:

**CS Foundations**
CS5's broad view:

CS 5 ~ all corners of CS

What *can* we compute...
... and *how well*?

Final Projects
- Picobot
- vPool
- TextClouds

CS Foundations
- biological analysis + algorithms
- recursion
- functions

CS Applications
- graphics
- variables
- loops

CS Theory
- media
- search and the web
- robots and computer vision
- circuits and memory
- data: classes and objects
"Theocomp"

(1) define "computer" precisely
(2) define "compute" precisely
(3) see what computers provably can't compute
(4) go back to step (1) and define things better...
(5) ... until time runs out...

CS Foundations  What can we compute... ... and how well?
"Theocomp"
computers ~ state machines

What is this state machine?
computers ~ state machines

What is this state machine?

Game-of-life State Machine
Unifying idea: **State**

The *state* of a computation is *all the internal information* needed to take the next step.

For Picobot, *next step is taken literally!*

```
0  NEWx  ->  S  1
```

previous state  external input  next step  next state
states ~ subtasks

State Machine:

each oval represents a different Picobot state

surroundings

state pattern -> move new state

0 x*** -> N 0
0 N*** -> X 1
1 ***x -> S 1
1 ***S -> X 0

starting funnel

the "go North" state

transitions move from state to state
Computation is a deliberate sequence of state-changes.

this doesn't seem very meaningful
Computation is a deliberate sequence of state-changes

Computer ~ a Finite State Machine

bits before
bits after
Finite state machine

State 0
- Transition on 0
- Transition on 1
- Start state
- Reject state

State 1
- Transition on 0
- Transition on 1
- Accept state
Finite state machine

an input \textbf{001011}

"where to go" transitions

Start state(s)

"input funnel"

accepting state(s)

do double circled
FSM: Finite state machine

mapping binary strings \( \Rightarrow \) booleans

What does each state MEAN?
What does this FSM do overall?

output for this input: Accept!
state: 0 001101
an input sequence always left-to-right

001011

transition on 0
transition on 0
transition on 0
transition on 1
transition on 1
transition on 1

State 0
State 1

Even # of 1s
Odd # of 1s
FSM: *Finite state machine*

**State 0**
- Transition on 0
- Transition on 1
- Transition on 1

**State 1**
- Transition on 0
- Transition on 1

an input sequence always left-to-right

001011

output for this input accepted!

"I've seen an ODD # of 1's"

"I've seen an EVEN # of 1's"

"I've seen an ODD # of 1's"

What does each state MEAN?
What does this FSM do overall?
 FSM:  *Finite state machine*

**State 0**
- Transition on 0
- Transition on 1
- Transition on 1

**State 1**
- Transition on 0
- Transition on 0

An input sequence: 0010111

Output for this input: rejected!

"I've seen an ODD # of 1's"

"I've seen an EVEN # of 1's"

What does each state MEAN?
What does this FSM do overall?
what does this state machine do?

empty string, '', or $\lambda$
graphical state-machine builder for hw12

JFLAP!
In general, what English phrase describes the rejected inputs?

This machine rejects strings with ... 

2 1s in a row

Extra Could fewer states produce the same accept-and-reject behavior here? What's the minimum #?

Hint: which strings have to be in separate states?
**Quiz**

Label each state with 1-2 inputs that "land" there...

In general, what English phrase describes the **rejected inputs**?

This machine rejects strings with ... two 1's in a row (anywhere in the string)

**Extra** Could fewer states produce the same accept-and-reject behavior here? What's the **minimum** #?

**Hint**: which strings **have** to be in separate states?

3 state smin.

0
1
11

add 1

add 0

add 0
In general, what English phrase describes the rejected inputs?

Extra Could fewer states produce the same accept-and-reject behavior here? What's the minimum #?

Hint: which strings have to be in separate states?
Extra Could fewer states produce the same accept-and-reject behavior here? What's the minimum #?

3 states min.

Hint: which strings have to be in separate states?

In general, what English phrase describes the rejected inputs?

This machine rejects strings with ...

two 1's in a row (anywhere in the string)

ends in zero 1's

ends in one 1

contains two (or more) 1's in a row

Label each state with 1-2 inputs that "land" there...

"graveyard" state: uses two transitions in JFLAP
FSMs are everywhere!

Mechanical vending machine

FSMs	are	everywhere!

- Penny, fifty cent piece, silver dollar, Canadian currency, CS 5 Euro, etc.
- Nickel, dime, quarter, 5 cents, 10 cents, 25 cents,

(some transitions not shown)

www.youtube.com/watch?v=85C4eh0mElg @ 1:42 !
FSMs are everywhere!
FSM ~ Game AI

The state-machine that controls Quake's *Shambler* monsters...

I'm *Quaking* in my AstroBoots
FSM ~ Game AI

Recognize this street?

Here, it's Ghost AI

https://www.youtube.com/watch?v=OfJN3yCFG8I
All robots use FSM control

What states can you "factor out" from watching this towel-folding?
Towel-folding states!

Fig. 2. The state machine model of the procedure: dashed lines indicate failure recovery cases. The images show an actual run.
Towel-folding?
singed out as a questionable use of dollars...
Towel-folding?

although everyone admits robots aren't good at towels!
**Build-your-own FSMs**

Draw a FSM accepting strings with at least two 1s (anywhere). Others are rejected.

Accepted examples: 0101, 00010110, 111011, 11

Rejected examples: 0100, 1000, 000000, 1, 0

*Hint* - modify this starter FSM by adding labels, transitions, and one more state:

```
1
↓
```

What's the minimum number of states needed?

Big picture!

Draw a FSM accepting strings in which the number of zeros (0s) is a multiple of 3, so there are 0, 3, 6, ... zeros. *1s don't matter!*

Accepted: 110101110, 11, 0000010

Rejected: 101, 0000, 111011101111

*Hint:* Is never change the state!

*Another hint:* make a triangle!

Extra! Draw a FSM accepting strings whose third-to-last digit (3d from the right) is a 1.

Accepted: 0100 and 01101

Rejected: 101001 and 11
Draw a FSM accepting strings in which the number of zeros (0s) is a multiple of 3, so there are 0, 3, 6, ... zeros. 

**Accepted:** 11011110, 11, 000010

**Rejected:** 101, 0000, 111011101111

**Hint:** 1s never change the state!

**Another hint:** make a triangle!

**FSMs ~ "software circuits"**

**Hint:** there are FIVE more transitions – but no more states - needed here
**Build-your-own FSMs**

Draw a FSM accepting strings with at least two 1s (anywhere). Others are rejected.

**Accepted examples:** 0101, 00010110, 111011, 11

**Rejected examples:** 0100, 1000, 000000, 1, 0

**Hint** - modify this starter FSM by adding labels, transitions, and one more state:

Draw a FSM that accepts strings that **don't** contain the pattern **110** anywhere.

**Accepted:** 1010001, 011

**Rejected:** 10100, 110

**Draw a FSM accepting strings in which the number of zeros (0s) is a multiple of 3, so there are 0, 3, 6, ... zeros. **1s don't matter!**

**Accepted:** 11010110, 11, 0000010

**Rejected:** 101, 0000, 11101101111

**Hint:** 1s never change the state!

**Another hint:** make a triangle!

**Draw a FSM accepting strings whose third-to-last digit (3d from the **left**) is a 1.**

**Accepted:** 1010001, 011

**Rejected:** 11000010, 11, 0

**Extra!** Draw a FSM accepting strings whose third-to-last digit (3d from the **right**) is a 1.

**Acc:** 0100 and 01101

**Rej:** 101001 and 11
Has at least two 1s... ?

Draw a FSM accepting strings with at least two 1s (anywhere). Others are rejected.

Accepted: 0101, 00010110, 111011, 11
Rejected: 0100, 1000, 000000, 1, 0

What do we need to complete this machine?
Has at least two 1s...?

Draw a FSM accepting strings with at least two 1s (anywhere). Others are rejected.

Accepted: 0101, 00010110, 111011, 11

Rejected: 0100, 1000, 000000, 1, 0

What do we need to complete this machine?
Number of 0s is div. by 3

Draw a FSM accepting strings in which the number of zeros (0s) is a multiple of 3, so there are 0, 3, 6, ... zeros. 1s don't matter.

Accepted: 110101110, 11, 0000010
Rejected: 101, 0000, 111011101111

Combined states:
- #Zeros%3 == 1
- #Zeros%3 == 0
- #Zeros%3 == 2
- #Zeros%3 == 3

Combine two of these?

Minimum number of states?
No occurrences of 110?

Draw a FSM accepting strings that do NOT anywhere contain the pattern 110

Accepted: 1010001, 0001011
Rejected: 101001100, 011001

Which transitions are still needed here?
Draw a FSM accepting strings in which the third digit (from the left) is a 1.

Accepted: 1010001 and 0110
Rejected: 1100100 and 11

Why must s1 and s2 be separate states?

Minimum number of states?
Third character is a 1

Draw a FSM accepting strings in which the third digit (from the left) is a 1.

Accepted: 1010001 and 0110
Rejected: 1100100 and 11

Why must s1 and s2 be separate states?

Minimum number of states?
Third-to-last character is a 1?

Draw a FSM accepting strings whose third-to-last digit (from the right) is a 1.

Accepted: 0100 and 01101

Rejected: 101001 and 11

Minimum number of states?
Third-to-last character is a 1

Do we need 15 states?

I don't accept this solution!
Something's not right here: it's down-right arrowing!
Third-to-last character is a 1

8 states?

8 states are required!
All robots use FSM control

... send me your FSM so that I can show it off in 2018!
Fig. 9. Situational Interpreter State Transition Diagram. All modes are sub-modes of the system RUN mode (Fig 4(b)).
FSMs driving robots...

MIT's car, Talos
FSMs driving robots...

MIT's car, Talos - *and its sensor suite*
State-machine *limits*?

Are there *limits* to what FSMs can do?

they can't necessarily drive safely...

But are there any *binary-string problems* that FSMs can't solve?
State-machine *limits*?

Let's build a FSM that accepts strings with *any* # of 0s followed by the *same* # of 1s.

- **Accepted:** 000111, 0011, 01, $\lambda$
- **Rejected:** 011, 001, 11100, 00110
State-machine *limits*?

Let's build a FSM that accepts strings with *any* # of 0s followed by the *same* # of 1s.

FSMs "can't count" at least, not arbitrarily high.
State-machines are limited.

*FSMs can't count*

at least not arbitrarily high...

We need a more powerful model than FSMs...

*What do we need to add?*
Thursday: Turing Machines

Lab sessions this week: State machines + final projects...