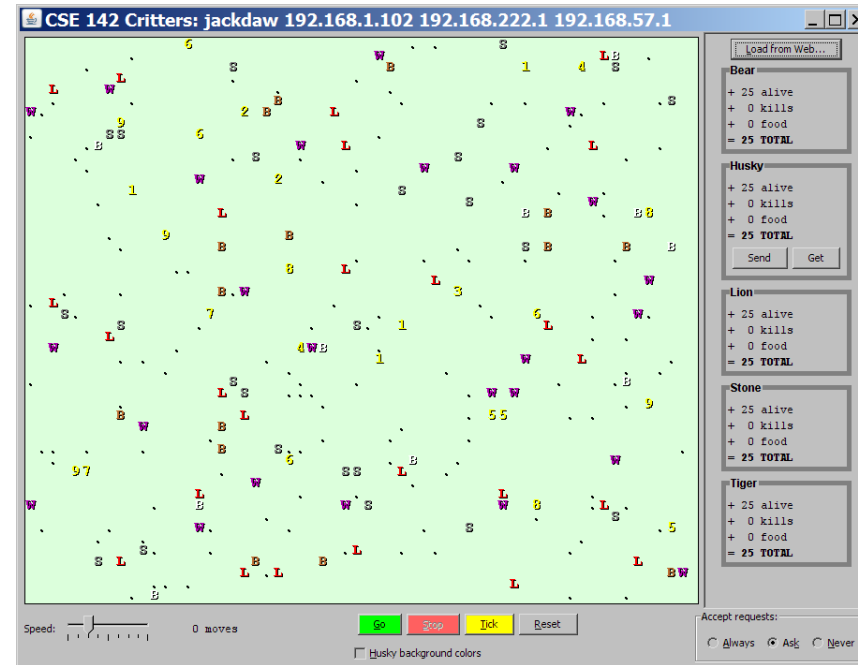# CSc 110, Autumn 2016

Lecture 32: Critters

Adapted from slides by Marty Stepp and Stuart Reges

CSc 110 Critters

- `Ant`
- `Bird`
- `Hippo`
- `Vulture`
- `WildCat`   (creative)

- behavior:
  - `eat`   eating food
  - `fight`   animal fighting
  - `get_color`   color to display
  - `get_move`   movement
 - `__str__`   letter to display

A Critter subclass

```
class name(Critter):

class Critter:
    def eat()
            # returns True or False
    def fight(opponent)
            # ROAR, POUNCE, SCRATCH
    def get_color()
            # returns a hex string
    def get_move()
            # returns NORTH, SOUTH, EAST, WEST, CENTER
    def __str__()
```

How the simulator works

- "Go" → loop:
  - move each animal (`get_move`)
  - if they collide, `fight`
  - if they find food, `eat`

- Simulator is in control!
  - `get_move` is <u>one move</u> at a time
    - (*no loops*)
  - Keep <u>state</u> (fields)
    - to remember future moves
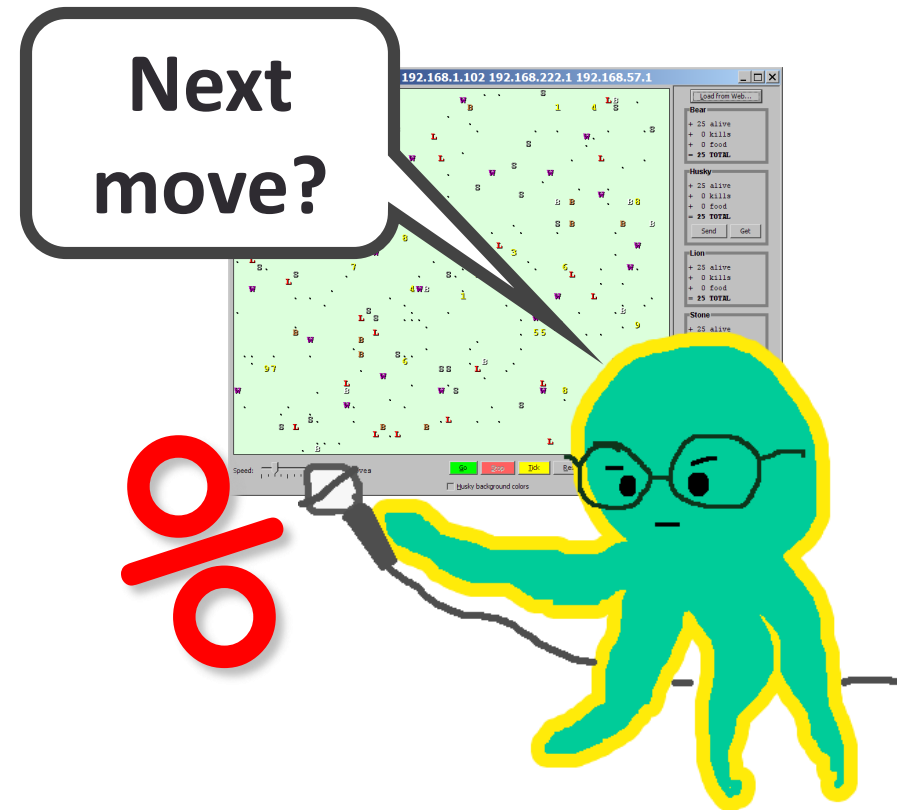
Development Strategy

- Simulator helps you debug
  - smaller width/height
  - fewer animals
  - **"Tick"** instead of "Go"

- Write your own main
  - call your animal's methods and print what they return

Critter exercise: `Cougar`

- Write a critter class `Cougar`:

Method	Behavior
`__init__`	
`eat`	Always eats.
`fight`	Always pounces.
`get_color`	Blue if the `Cougar` has never fought; red if he has.
`get_move`	Walks west until he finds food; then walks east until he finds food; then goes west and repeats.
`__str__`	`"C"`

Ideas for state

- You must not only have the right state, but update that state properly when relevant actions occur.

- Counting is helpful:
  - How many total moves has this animal made?
  - How many times has it eaten? Fought?

- Remembering recent actions in fields is helpful:
  - Which direction did the animal move last?
    - How many times has it moved that way?
  - Did the animal eat the last time it was asked?
  - How many steps has the animal taken since last eating?
  - How many fights has the animal been in since last eating?

Critter exercise: `Anteater`

- Write a critter class `Cougar`:

Method	Behavior
`__init__`	
`eat`	Eats 3 pieces of food and then stops
`fight`	randomly chooses between pouncing and roaring
`get_color`	pink if hungry and red if full
`get_move`	walks up two and then down two
`__str__`	`"a"` if hungry `"A"` otherwise