

# Data, Math and Methods

## Week 5, State Machines Project



# Today

- State Machines Project
  - What we learned about State Machines into a project

# State Machines in code

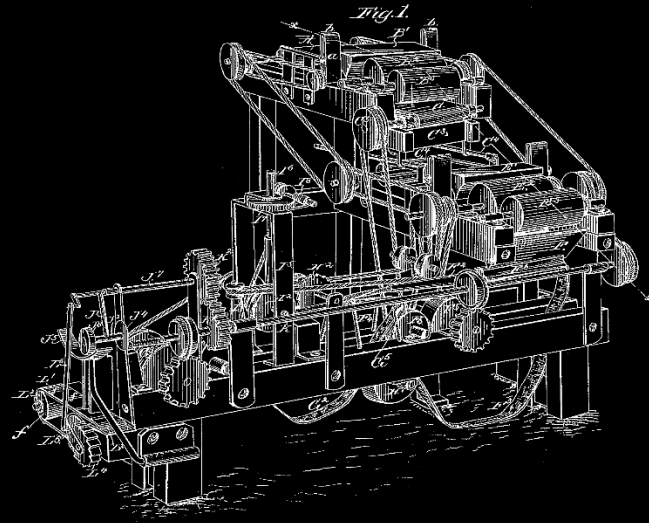
- Let's code up a simple state machine
- State can control an output – for example color / sound sample in the loop / displayed image

# Machines

- Repetition from the last class:

## Inputs >

*In some form that the machine will understand  
= we call that an alphabet*



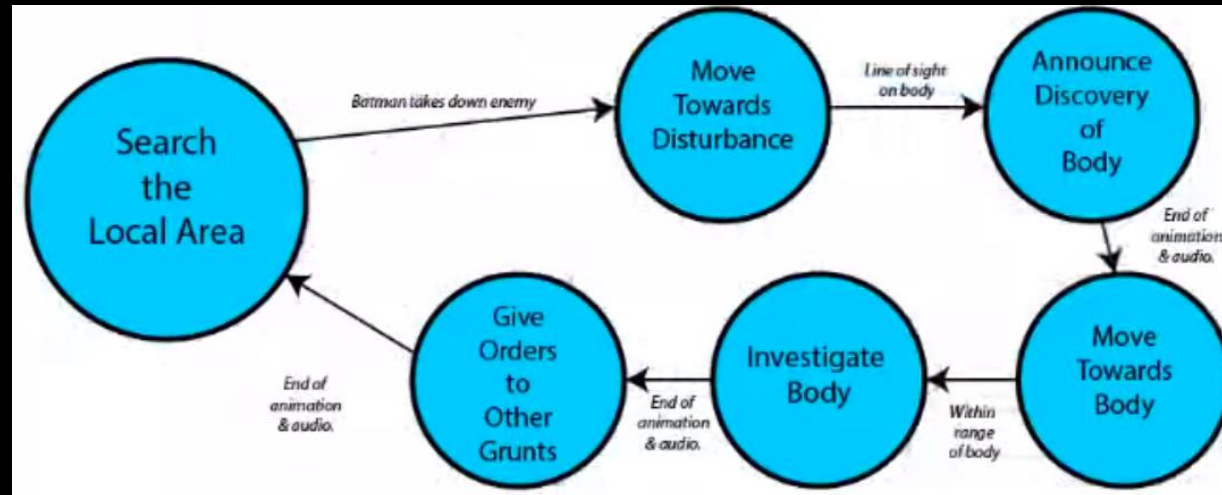
## > Outputs

*Again some which we are expecting*

Machine has a state (turned on / off / waiting for input / etc ...)

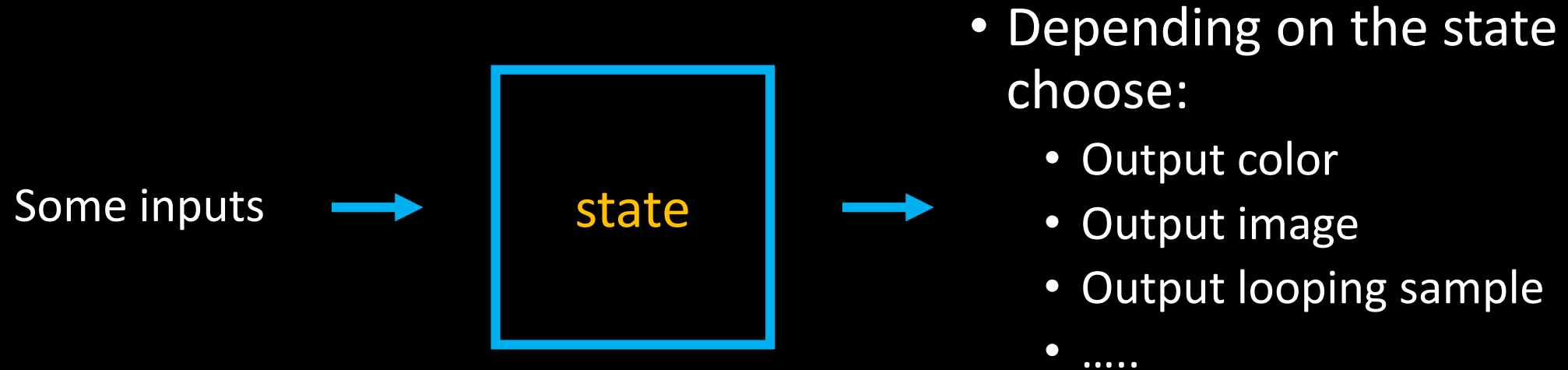
# State Machines uses

- So for example as AI in Games:



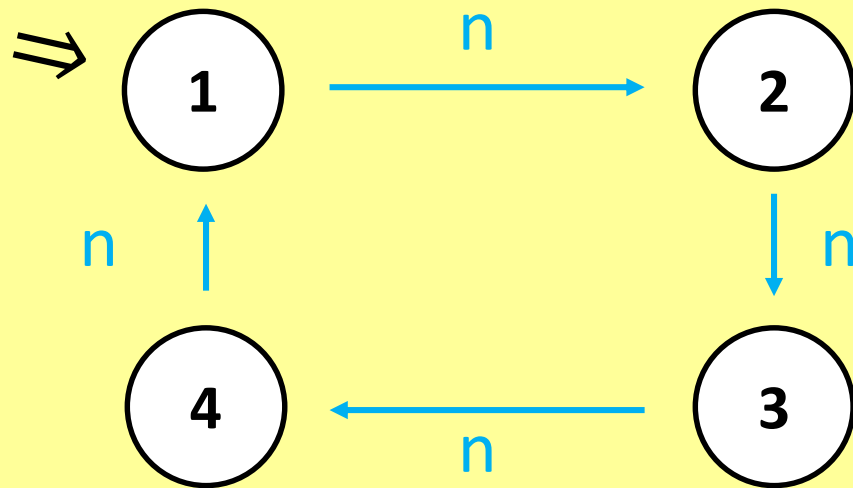
# State Machines

- In our coding scenario we want this:



# State Machines example

What does this machine do?

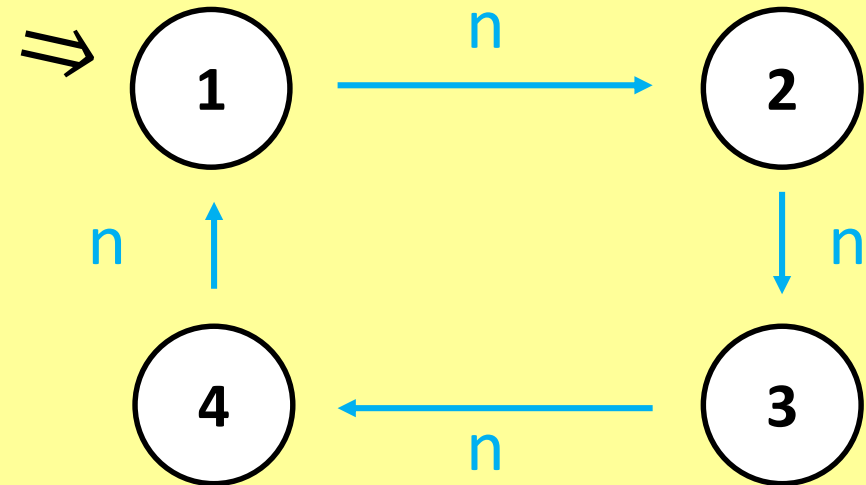


It loops through states (1 to 4) when pressing “n”.

In this case we don't have an “accepting” state, as an output we can read which state we are on

# State Machines formalized:

- Set of states (1,2,3,4)
  - Start: starting state (1)
- Accepting alphabet ("n")
- Transitions between states
  - 1  $\Rightarrow$  2 with "n"
  - 2  $\Rightarrow$  3 with "n"
  - ... etc
- **Mapping** between the state number and outputted color (for example)
  - 1  $\Rightarrow$  red, 2  $\Rightarrow$  orange, 3  $\Rightarrow$  green, ...





# Today's coding task

- Build a state machine inside the rest of the code which controls the behavior
- We get some inputs (for example key presses) and in the end we want to map the machine state onto a output (a color, or a sound sample being played)
- (Important!) Allow your code to work with any state machine – don't hardcode just one behavior, but let me change the control state machine mechanism to (in theory) any possible state machine!

# Homework

- Work on this task and have something to show me in the next class
- Include your drawing of the state machine on paper (design of the logic of behavior of it) when showing me.
- Ideally: machines with more than 5 states (unless there's a smart reason why you only need less :D)