

# CS 4700/5700

# Foundations of

# Artificial Intelligence

Cornell University  
Instructor: Kevin Ellis  
Adapted from Berkeley CS188

# What is artificial intelligence

What can AI do?

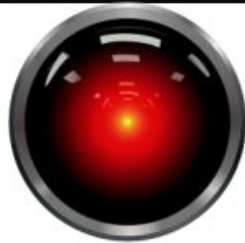
What should we worry about?

What can we do about these things?

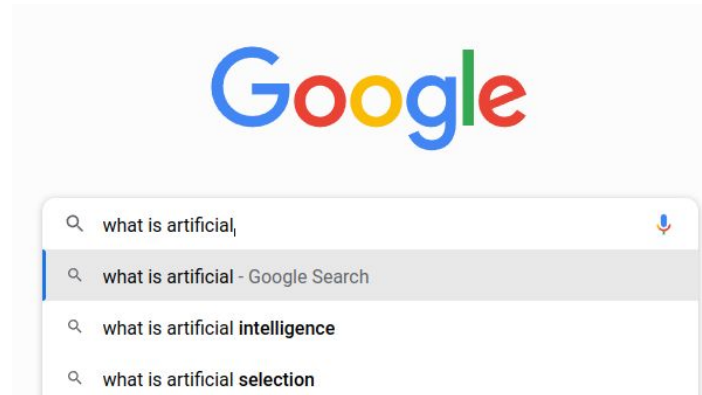
What should we not worry about?

What is this course?

# AI in Science Fiction



# AI in reality



# Rational Decisions

We use the term rational in a very specific technical way:

- Rational: maximally achieving pre-defined goals

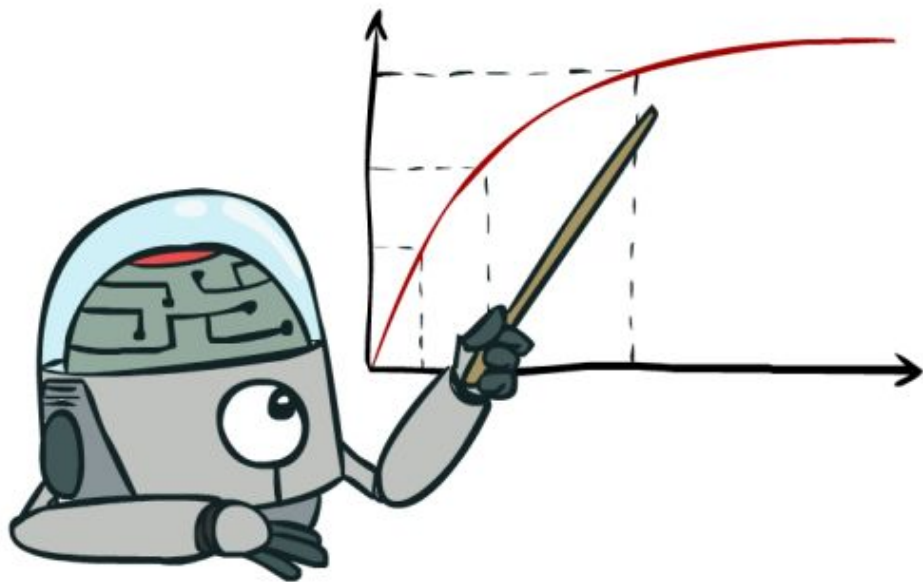
- Rationality only concerns what decisions are made

  - (not the thought process behind them)

- Goals are expressed in terms of the utility of outcomes

Being rational means maximizing your expected utility

# Maximize Your Expected Utility



# What about the brain?

Brains are surprisingly good at making close to optimal decisions

Reverse engineering the mind (cognitive science) is hard (but not impossible)

“Brains are to intelligence as wings are to flight”

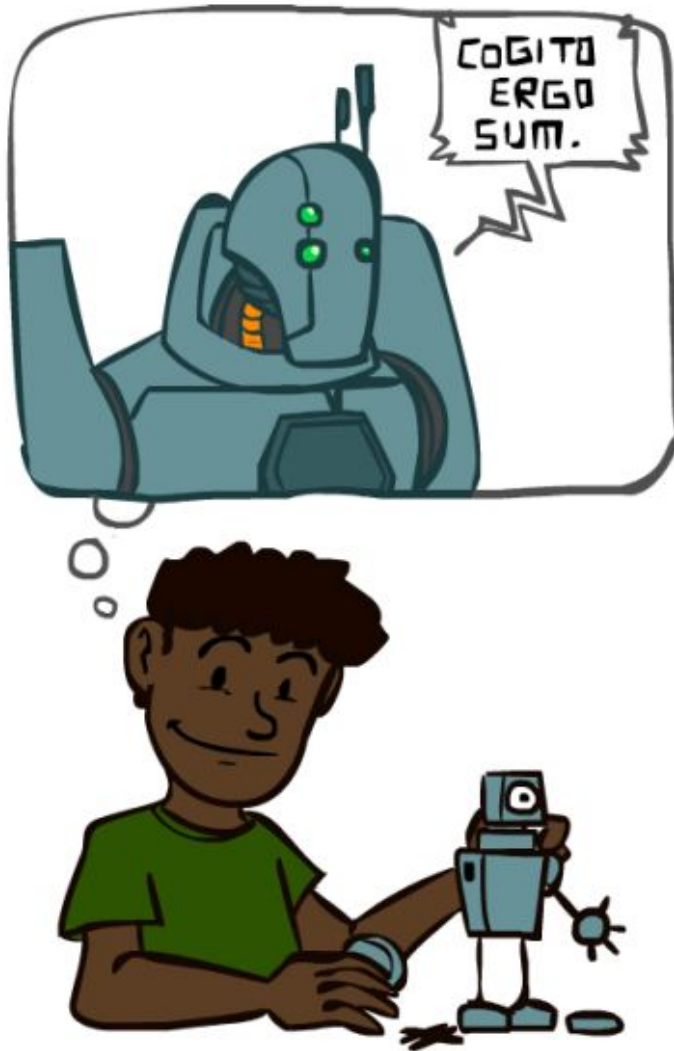
Lessons learned from the brain:

- Learning

- Simulation

- Dealing with uncertainty

# AI History





# AI history

1940-1950: Early days

1943: McCulloch & Pitts: Boolean circuit model of brain

1950: Turing's "Computing Machinery and Intelligence"

1950—70: Excitement: Look, Ma, no hands!

1950s: Early AI programs, including Samuel's checkers program, Newell & Simon's Logic Theorist, Gelernter's Geometry Engine

1956: Dartmouth meeting: "Artificial Intelligence" adopted

1965: Robinson's complete algorithm for logical reasoning

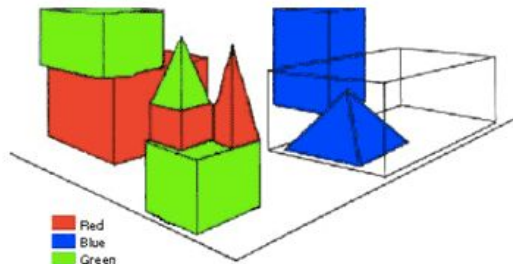
# AI history

1970—90: Knowledge-based approaches

1969—79: Early development of knowledge-based systems

1980—88: Expert systems industry booms

1988—93: Expert systems industry busts: “AI Winter”



Shrdlu, CYC, ...

**Person:** Pick up a big red block.

**Computer:** OK.

**Person:** Grasp the pyramid.

**Computer:** I don't understand which pyramid you mean.

# AI history

1990—: Statistical approaches

Resurgence of probability, focus on uncertainty

General increase in technical depth

Agents and learning systems... “AI Spring”?

1996: Kasparov defeats Deep Blue at chess

1997: Deep Blue defeats Kasparov at chess



“I could feel --- I could smell ---  
a new kind of intelligence  
across the table.” ~Kasparov

# AI History

2000—: Where are we now?

Big data, big compute, neural networks

Some re-unification of sub-fields

AI used in many industries

Chess engines on ordinary laptops can defeat top chess players

2011: IBM's Watson defeats Ken Jennings and Brad Rutter at Jeopardy!

2016: DeepMind's AlphaGo beats Lee Sedol at Go



# What can AI do today?

yes      Play a decent game of Jeopardy?

yes      Win against any human at chess?

yes      Win against the best humans at Go?

yes      Play a decent game of tennis?

yes      Grab a particular cup and put it on a shelf?

no        Unload any dishwasher in any home?

yes      Write boring software?

maybe    Discover new algorithms from scratch?

# What can AI do today?

sorta      Drive safely along the highway?

no          Drive **safely** to Syracuse?

yes        Buy a week's worth of groceries on the web?

no        Buy a week's worth of groceries at Wegmans?

only  
boring    Discover and prove a new mathematical theorem?

ones  
no        Perform a surgical operation?

yes        Translate spoken Chinese into spoken English in real time?

# Natural Language

## Speech technologies

- Automatic speech recognition (ASR)

- Text-to-speech synthesis (TTS)

## Dialog systems

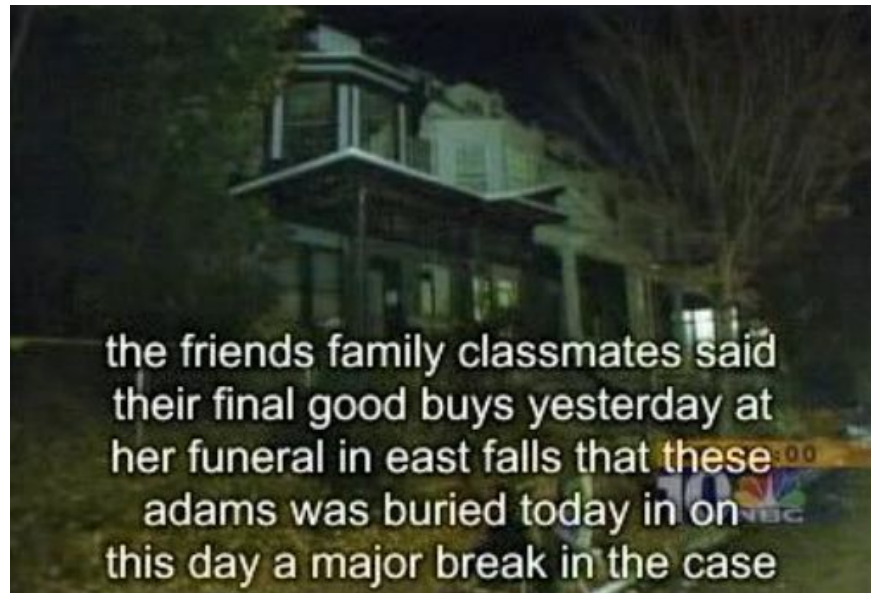
- Language processing technologies

- Question answering

- Machine translation

- Web search

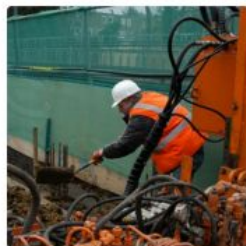
- Text classification, spam filtering, etc...



# Computer Vision



"man in black shirt is playing guitar."



"construction worker in orange safety vest is working on road."



"two young girls are playing with lego toy."



"boy is doing backflip on wakeboard."



"girl in pink dress is jumping in air."



"black and white dog jumps over bar."



"young girl in pink shirt is swinging on swing."



"man in blue wetsuit is surfing on wave."



# Should I take CS4700?

Yes, if you want to know how to design rational agents!

CS4700 gives you extra mathematical maturity

CS4700 gives you a survey of other non-CS fields that interact with AI (e.g. robotics, cognitive science, economics)

No, if you want Machine Learning (CS4780)!

Disclaimer: If you're interested in making yourself more competitive for AI jobs, machine learning (CS4780) is a better fit.

# CS4700 is about Reasoning + Decision Making

Given what I know,

and what I observe right now,

what should I believe?

and what should I do?

# Designing Rational Agents

An agent is an entity that perceives and acts.

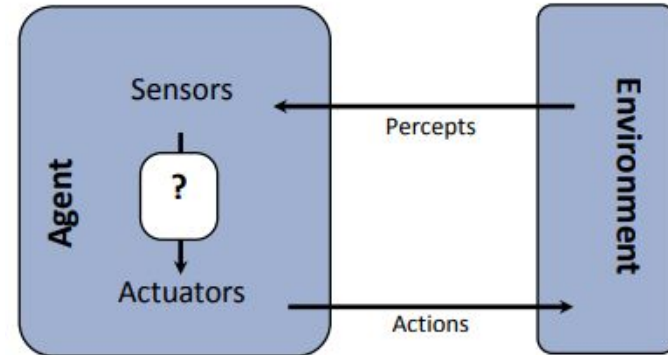
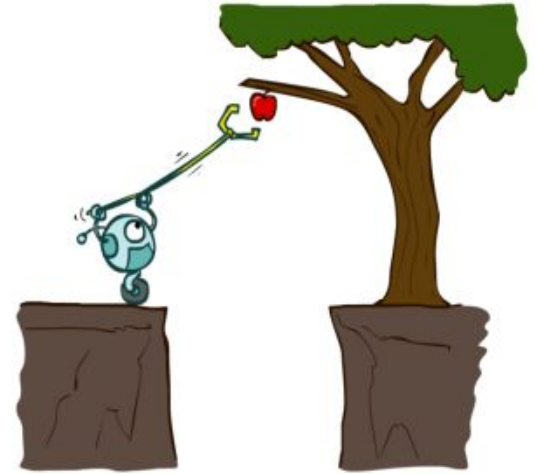
A rational agent selects actions that maximize (expected) utility.

Characteristics of the percepts, environment, and action space dictate techniques for selecting rational actions

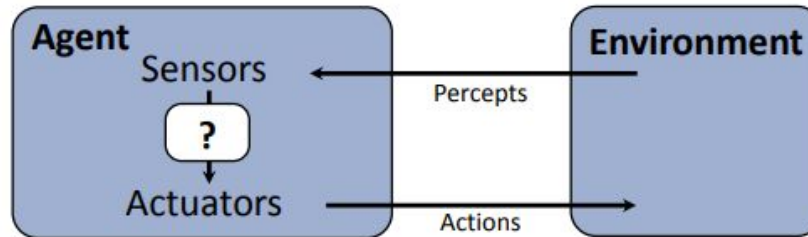
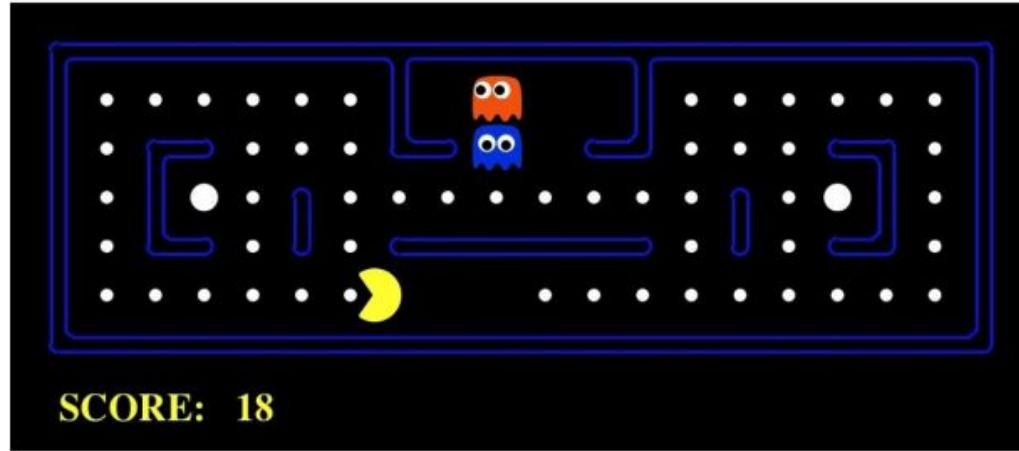
This course is about:

- General AI techniques for a variety of problem types

- Learning to recognize when and how a new problem can be solved with an existing technique



# Pac-Man as a rational agent



# Your homework for Wednesday: Play PacMan

Google

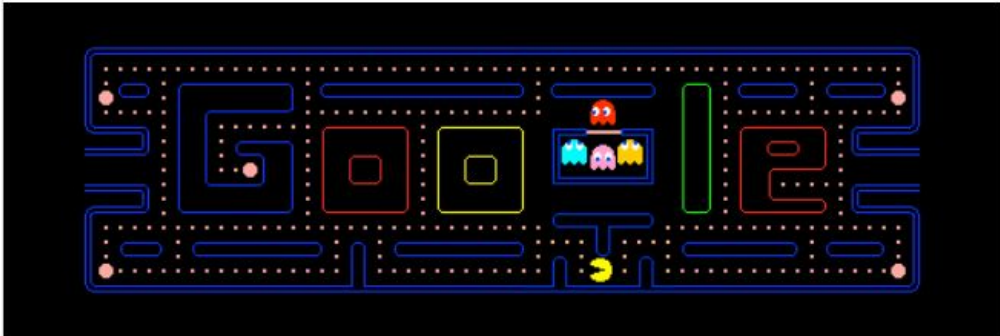
pacman online game

✕ 🔊 📷 🔍

🔍 All 📺 Videos 🖼 Images 🛒 Shopping 📰 News ⋮ More Tools

About 16,000,000 results (0.47 seconds)

PAC-MAN Doodle

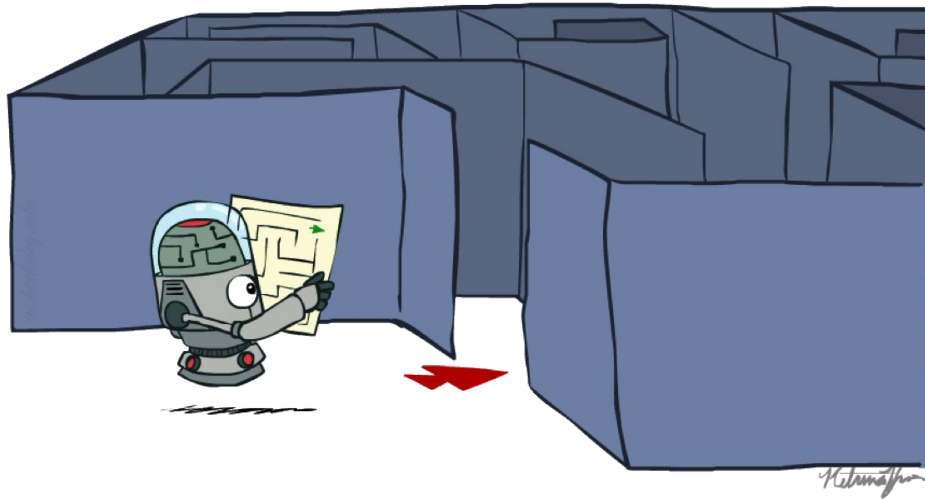


The image shows the Pac-Man Doodle game interface. It features a black rectangular play area with a blue maze. The maze is shaped like the word 'GOLE' in a stylized font. The letter 'G' is on the left, 'O' is in the middle, 'L' is on the right, and 'E' is on the far right. The Pac-Man character, a yellow circle with a wedge missing, is at the bottom center of the maze. Several red dots (Pac-Man's food) are scattered throughout the maze. In the center of the maze, there are three colorful ghosts (red, blue, and yellow) and a small yellow Pac-Man character. Below the play area is a blue button with the word 'Play' in white. At the bottom of the page, there is a small copyright notice: 'PAC-MAN™ & ©1980 BANDAI NAMCO Entertainment Inc.'

Play

PAC-MAN™ & ©1980 BANDAI NAMCO Entertainment Inc.

# Search



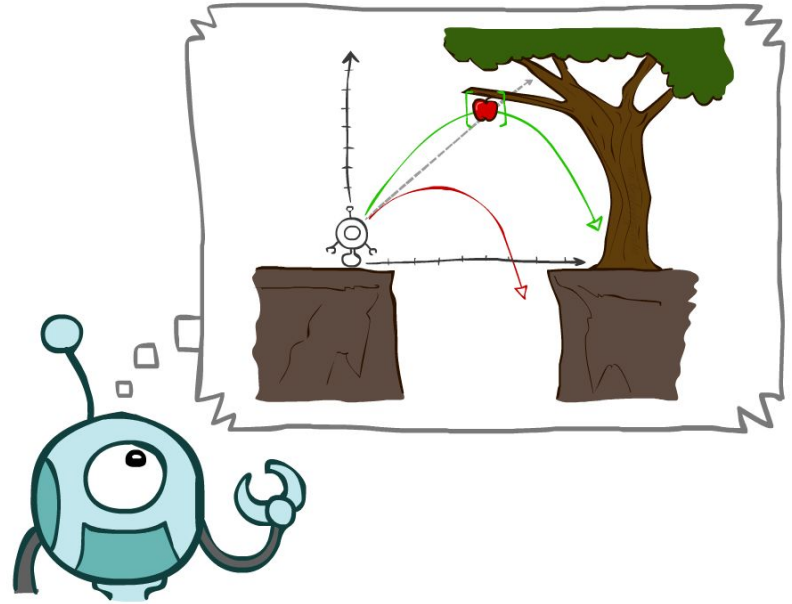
Instructor: Kevin Ellis

Cornell University

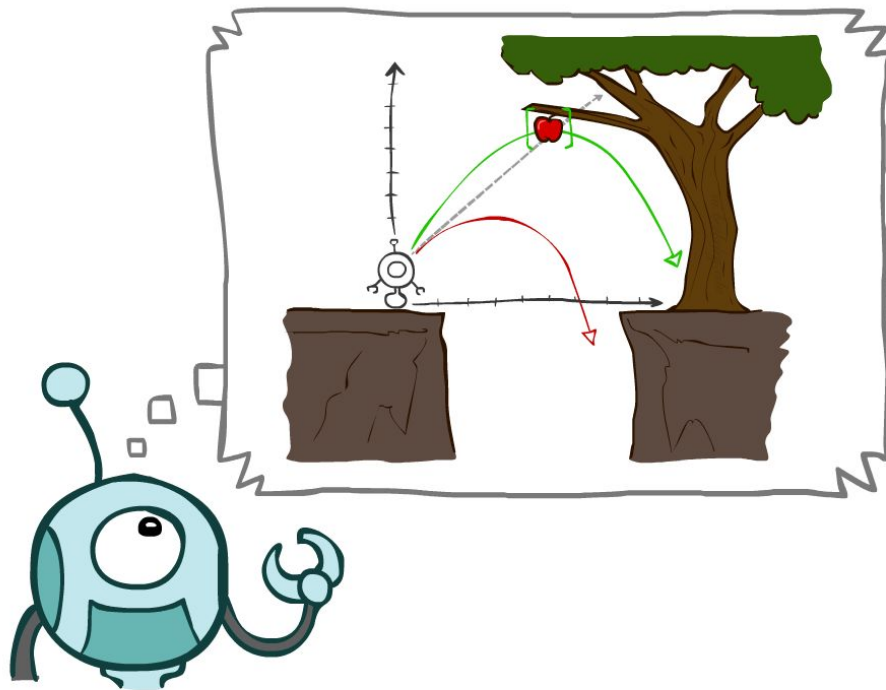
[These slides were created by Dan Klein and Pieter Abbeel for CS188 Intro to AI at UC Berkeley (ai.berkeley.edu).]

# Today

- Agents that Plan Ahead
- Search Problems
- Uninformed Search Methods
  - Depth-First Search
  - Breadth-First Search
  - Uniform-Cost Search



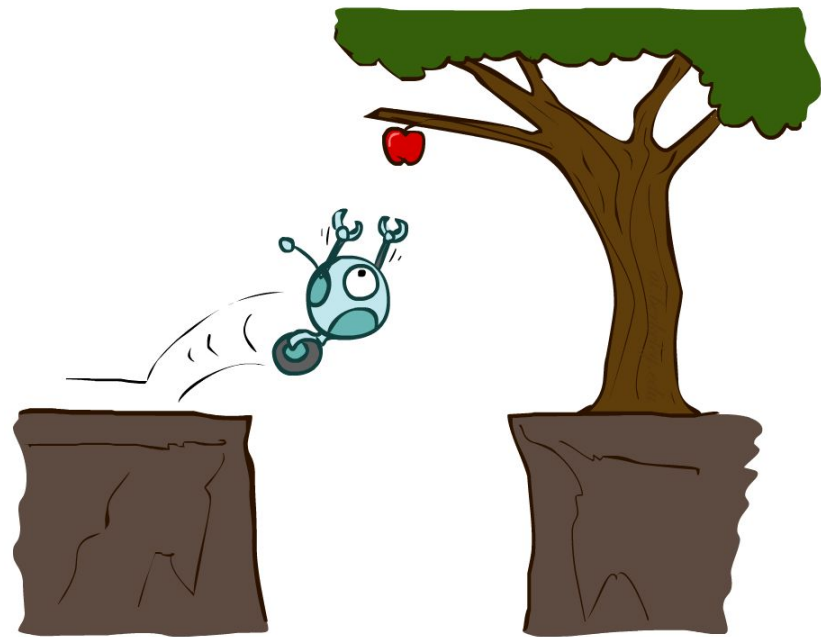
# Agents that Plan





# Reflex Agents

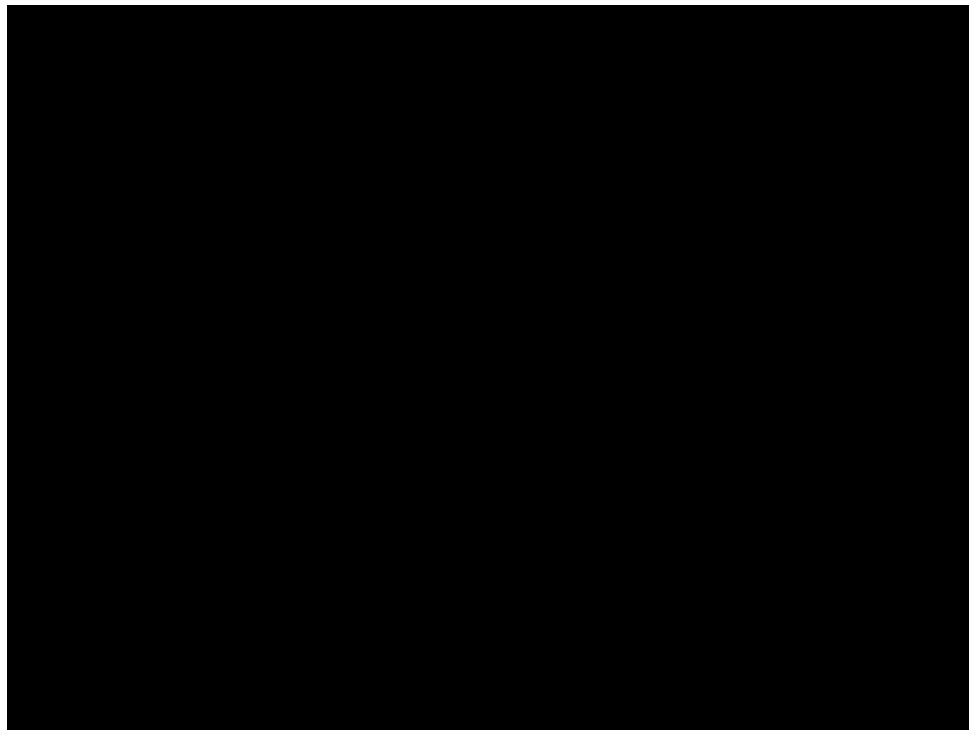
- Reflex agents:
  - Choose action based on current percept (and maybe memory)
  - May have memory or a model of the world's current state
  - Do not consider the future consequences of their actions
  - Consider how the world IS
- Can a reflex agent be rational?



[Demo: reflex optimal (L2D1)]

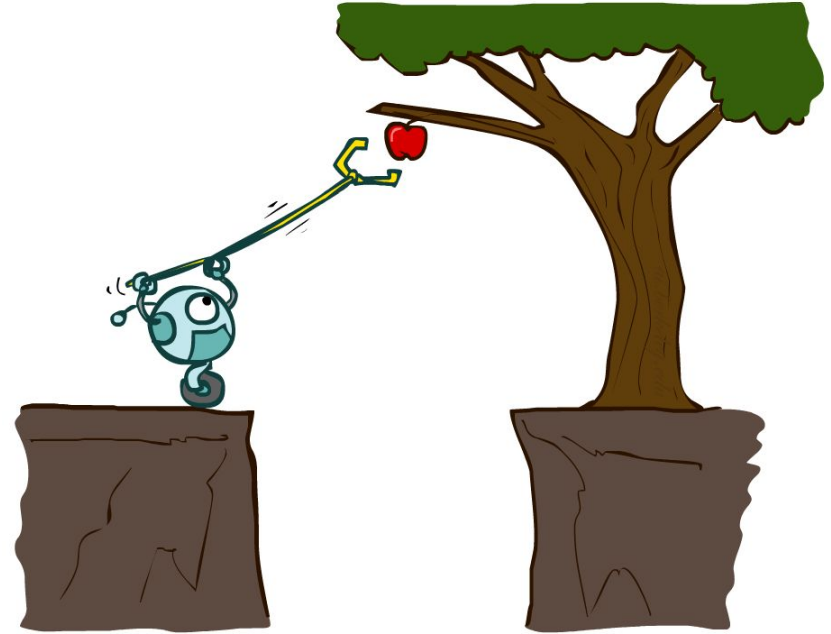
[Demo: reflex optimal (L2D2)]

# Video of Demo Reflex Optimal



# Planning Agents

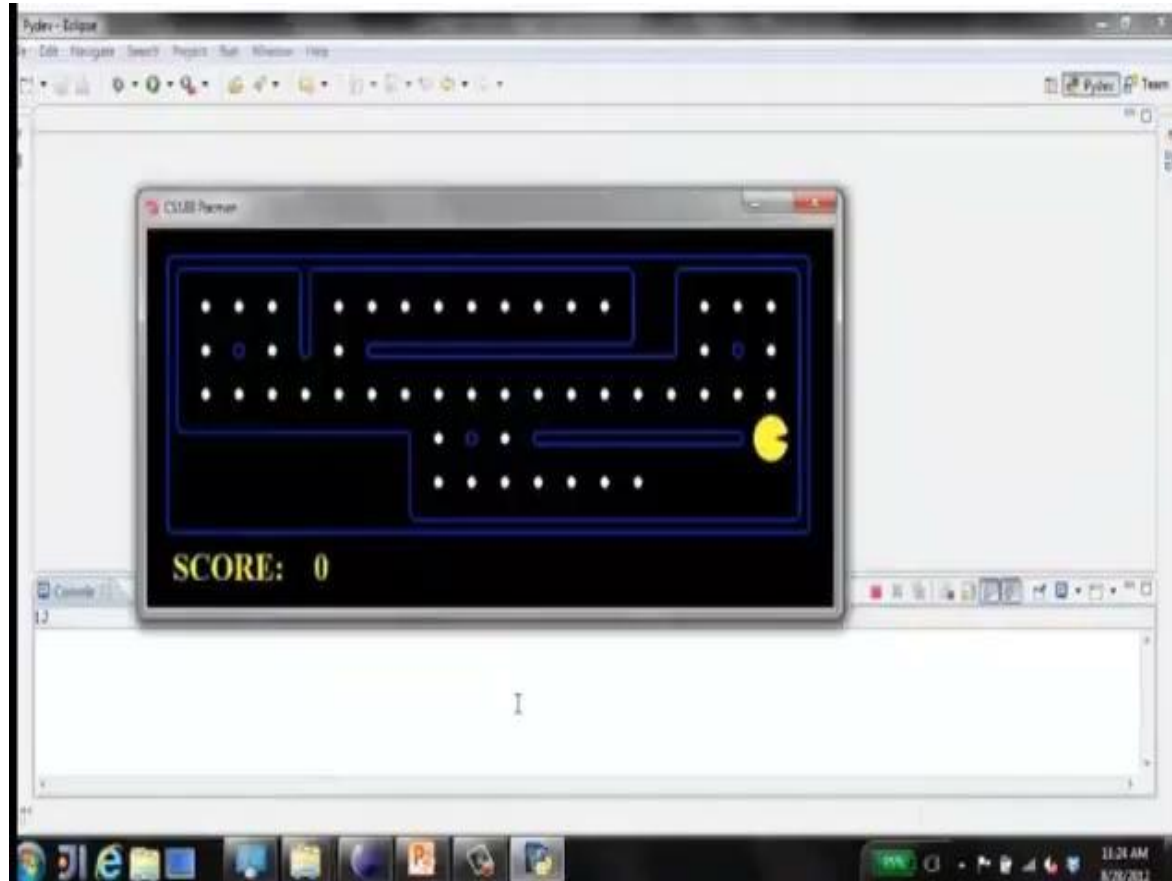
- Planning agents:
  - Ask “what if”
  - Decisions based on (hypothesized) consequences of actions
  - Must have a model of how the world evolves in response to actions
  - Must formulate a goal (test)
  - **Consider how the world WOULD BE**
- Optimal vs. complete planning
- Planning vs. replanning



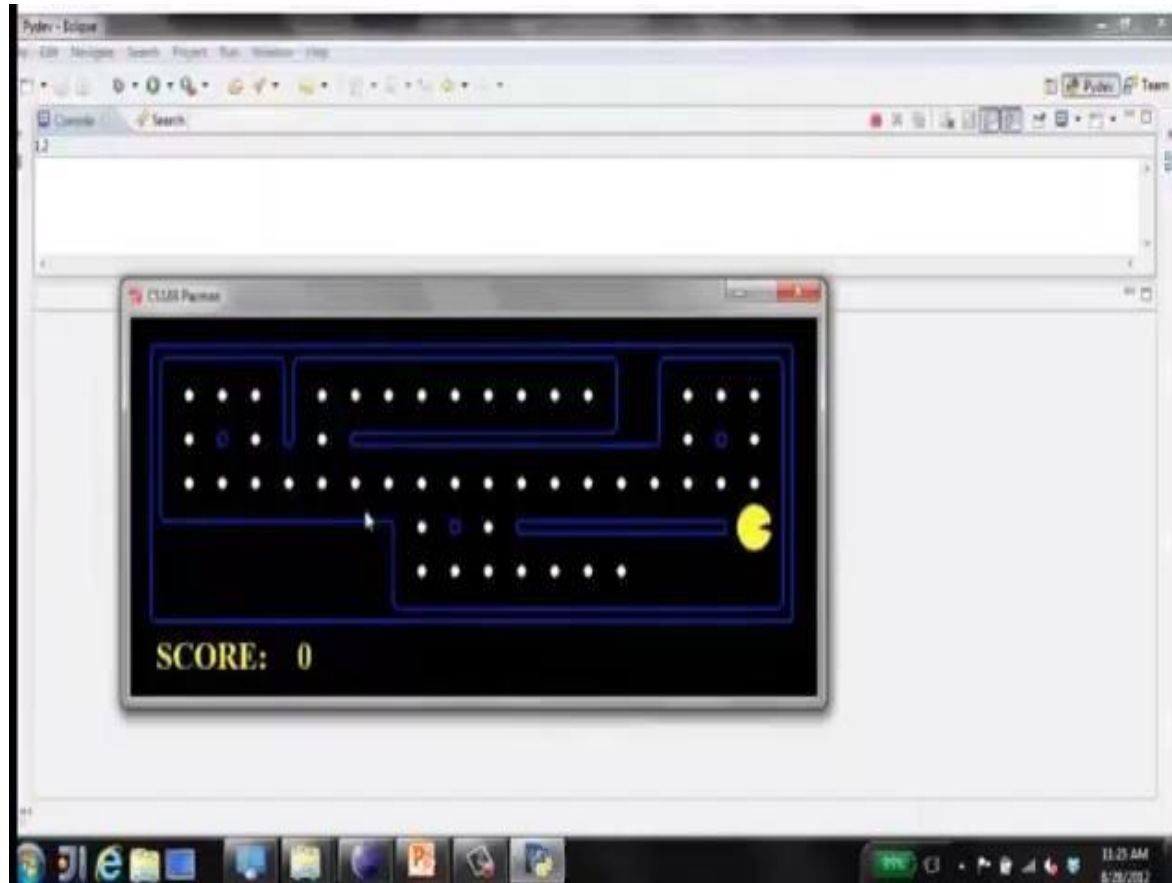
[Demo: re-planning (L2D3)]

[Demo: mastermind (L2D4)]

# Video of Demo Replanning



# Video of Demo Mastermind



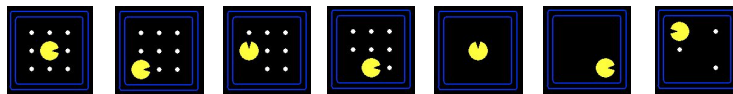
# Search Problems



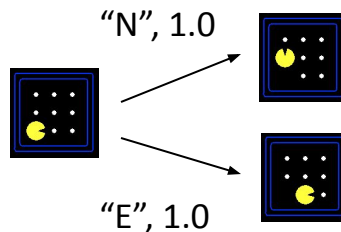
# Search Problems

- A **search problem** consists of:

- A state space



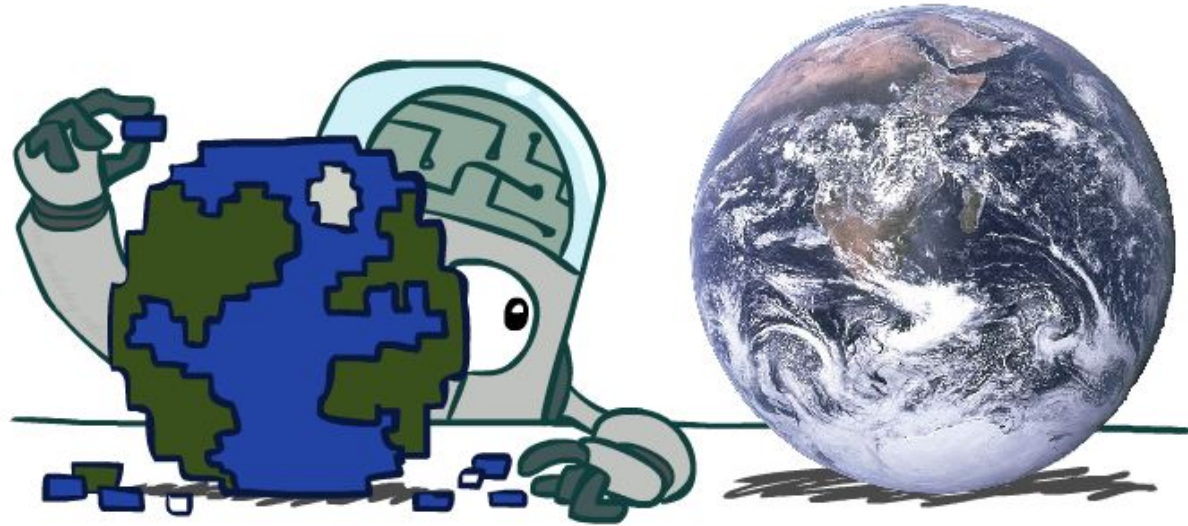
- A successor function  
(with actions, costs)



- A start state and a goal test

- A **solution** is a sequence of actions (a plan) which transforms the start state to a goal state

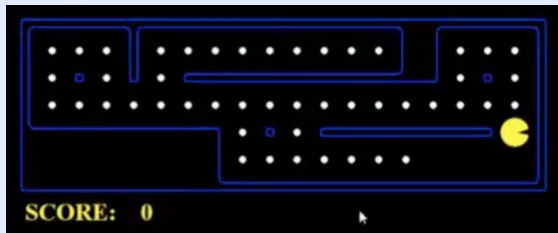
# Search Problems Are Models





# What's in a State Space?

The **world state** includes every last detail of the environment

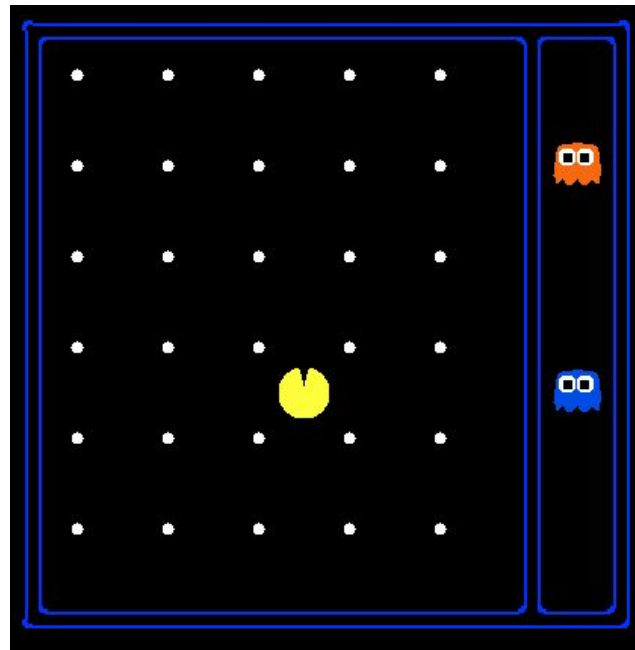


A **search state** keeps only the details needed for planning (abstraction)

- Problem: Pathing
  - States:  $(x,y)$  location
  - Actions: NSEW
  - Successor: update location only
  - Goal test: is  $(x,y)=END$
- Problem: Eat-All-Dots
  - States:  $\{(x,y), \text{dot booleans}\}$
  - Actions: NSEW
  - Successor: update location and possibly a dot boolean
  - Goal test: dots all false

# State Space Sizes?

- World state:
  - Agent positions: 120
  - Food count: 30
  - Ghost positions: 12
  - Agent facing: NSEW
- How many
  - World states?  
 $120 \times (2^{30}) \times (12^2) \times 4$
  - States for pathing?  
120
  - States for eat-all-dots?  
 $120 \times (2^{30})$



To Be Continued

On to Logistics

# Enrollment

Contact: [cs-course-enroll@cornell.edu](mailto:cs-course-enroll@cornell.edu)

<https://courses.cis.cornell.edu/courses/cs4700/2024sp/>

## Enrollment Questions

If you have a question about getting into the course, please contact the CS Enrollment Administrator at [cs-course-enroll@cornell.edu](mailto:cs-course-enroll@cornell.edu). Most questions about enrollment are not specific to this course and you'll find the answers at:

- [Enrollment information for CS classes](#)
- [Waitlist FAQs](#).
- [Information specifically for 4000- and 5000-level courses](#).



Lectures

MW 2:55-4:10pm

**In person** Hollister Hall B14

# Office Hours

<https://courses.cis.cornell.edu/courses/cs4700/2024sp/>

## Course Staff:

- Instructors:
  - [Prof. Kevin Ellis](#), Gates 442B
- CS Enrollment Administrator:
  - [Megan Gatch](mailto:cs-course-enroll@cornell.edu), [cs-course-enroll@cornell.edu](mailto:cs-course-enroll@cornell.edu)
- Course Administrator:
  - [Lacy Jordaens](mailto:Lacy Jordaens, CS4700-L@cornell.edu), [CS4700-L@cornell.edu](mailto:CS4700-L@cornell.edu)
- Graduate TAs:
  - [Vivian Nguyen](#)
  - [Top Piriyakjulkj](#)
  - [Jinyan Su](#)

- Office Hours can be found at <https://calendar.google.com/calendar/u/0?cid=ZWU2ZGNjOTRlZTmNjk5NDA2MjFiZjE2NmY2Mjg3YTQyODImMTI1MTJlMzRkODM5OGExNDJjYWE3NmE2ODc4ZUBncm91cC5jYWxlbnRhci5nb29nbGUuY29t.>

# 4700 Online

Webpage (everything): <https://courses.cis.cornell.edu/courses/cs4700/2024sp/>

Canvas (getting homework): <https://canvas.cornell.edu/courses/63132>

Gradescope (submitting homework): <https://www.gradescope.com/courses/697381>

Ed Discussions (asking questions): <https://edstem.org/us/courses/53763/discussion/>

# Grading

Prelim: 35%

Final: 40%

Cumulative

Homework: 25%

6 python programming assignments

Autograded



## Letter grade cutoffs

90%+ is some kind of A

80%+ is some kind of B

70%+ is some kind of C

## Letter grade cutoffs

90%+ is ***at least*** some kind of A

80%+ is ***at least*** some kind of B

70%+ is ***at least*** some kind of C

If the class average is unexpectedly low, we will lower these grade cutoffs.

If the class average is unexpectedly high, we will ***not*** raise these grade cutoffs.

# Late policy

Penalty of 3% per day

Example: You get 90% on the homework (raw score)

You submit it 1.4 days late, rounds up to 2: penalty 3%/day \* 2 days

Adjusted score:  $(1 - 3\%/day * 2 \text{ days}) * 90\% = 94\% * 90\% = 84.6\%$

Extenuating circumstances:

Contact course administrator, Lacy Jordaens, [CS4700-L@cornell.edu](mailto:CS4700-L@cornell.edu)

# Exam scheduling

Prelim:

Thursday, March 14, 7:30pm

All exam conflicts should be reported to us using a form that we will send out a month before the prelim (February 14). Please include details on the other exam or course causing the conflict.

# Prerequisites

Trees and graphs and algorithms on them

Probability

Propositional and first-order logic

Python

## Laptop policy

Laptops only to the back and sides of the room

# What's different about 4700

The new 4700 is based on CS188 at Berkeley:

<https://inst.eecs.berkeley.edu/~cs188/fa18/index.html>

The new 4700 will not cover machine learning

...except for reinforcement learning

Pro: Great programming homework!

Pro: Great slides!

Con: Less ML

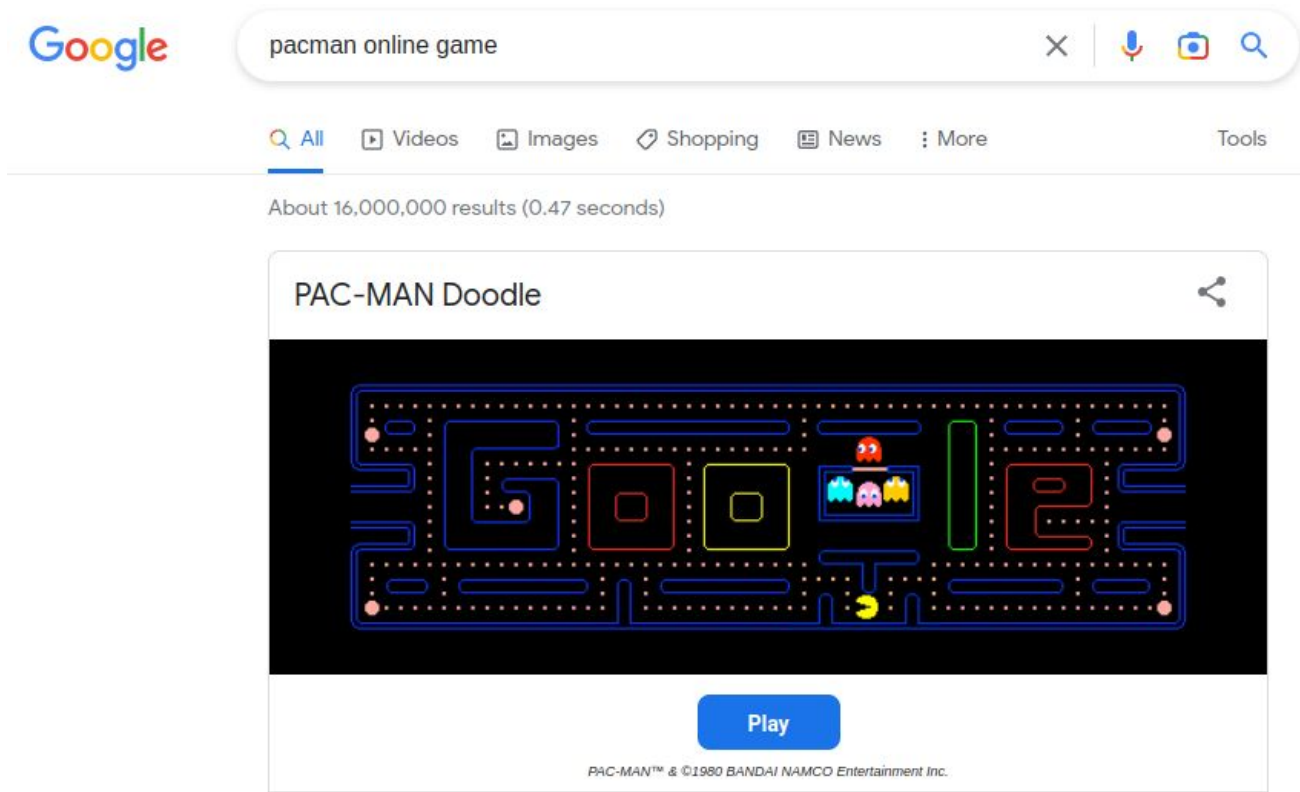
Con: Relatively easy to cheat on homework

# Why you should not violate academic integrity

1. The homework is (relatively) fun
2. The homework does not count for much of your grade
  - a. 25%
3. We will absolutely pursue the strongest possible consequences available to us for any academic integrity violations surrounding the programming homework
4. Way, way better to lose a few points on something that isn't worth much of your grade, compared to getting caught cheating on the programming assignments



# Your homework for Wednesday: Play PacMan



# Also your homework for Wednesday: Install dependencies

## Python environment/dependencies #1



Kevin Ellis **STAFF**  
Yesterday in **General**



UNPIN



STAR



WATCHING

59

VIEWS



6

To make sure you don't get any python issues with the homework, please do the following. Doing this ensures that you have all of the python dependencies installed, and also that your code can be handled by the gradescope autograder.

1. Install Anaconda, a system which manages different python installations and packages.