# A friendly reminder!

**Make sure that you complete the CSCI S-80 software form**

# Section with Barbara

# Week 0

Introduction

Framework

AI Stories

Projects

Getting Set Up

Helpful Textbook

MY BACKGROUND:

**STATISTICIAN**

MY TEACHING QUALIFICATION:

**FORMER STUDENT OF THIS CLASS**

MY MEASURE OF SUCCESS:

**BEGINNER FRIENDLY**

# About Me

“

# Intelligence measures an agent’s ability to achieve goals in a wide range of environments

Universal Intelligence: A Definition of Machine Intelligence. Legg, S., & Hutter, M. (2007)

”

# About S-80: Subject Matter

## SYMBOLIC AI

EXPERT SYSTEMS + RULES

WEEK 0: SEARCH

WEEK 1: KNOWLEDGE

WEEK 3: OPTIMIZATION

## COMPUTATIONAL AI

TRAINING + OBSERVATION

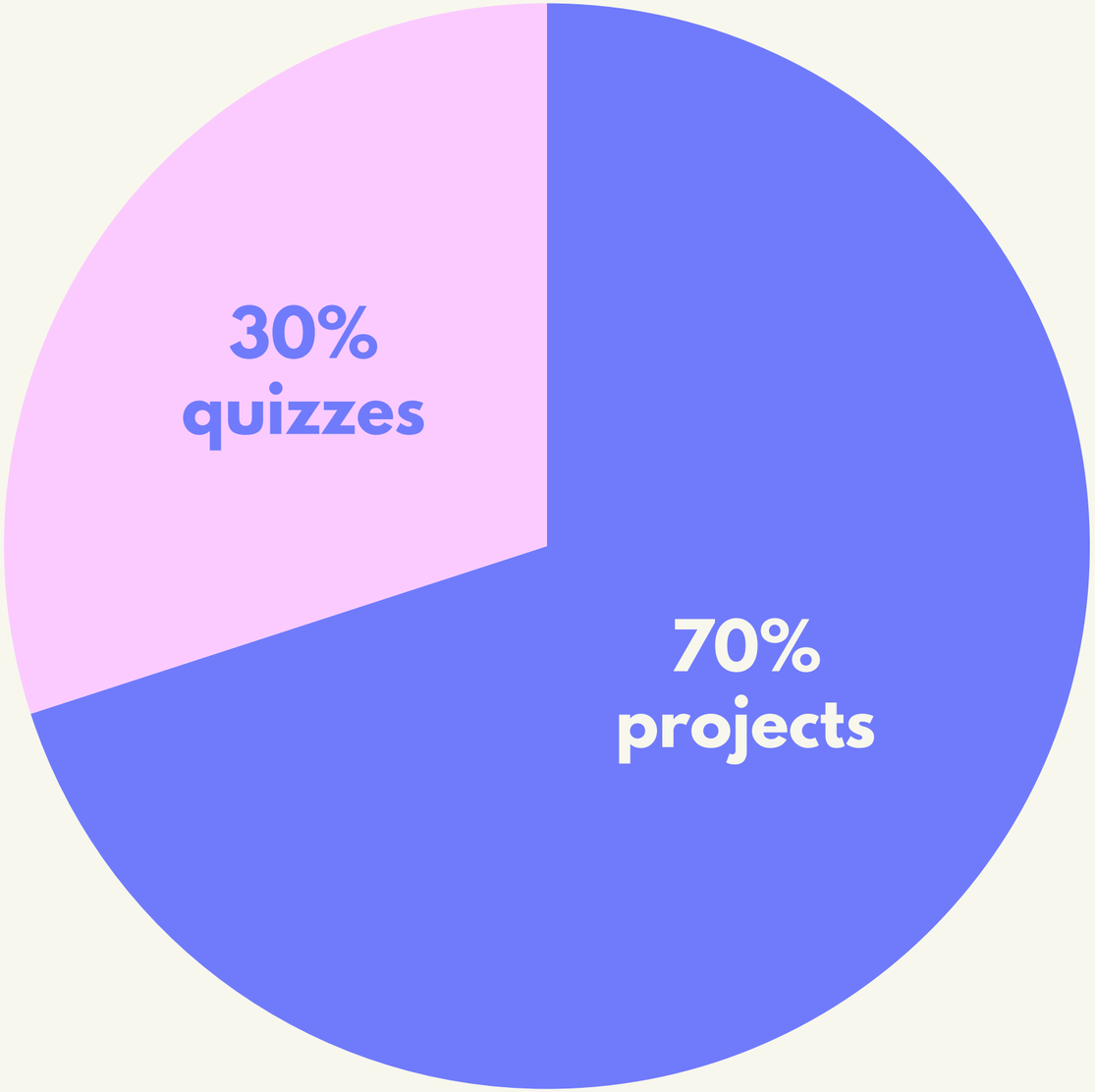
WEEK 2: UNCERTAINTY

WEEK 4: LEARNING

WEEK 5: NEURAL NETWORKS

WEEK 6: LANGUAGE

# About S-80: Grades



## About Section

DURATION:

**90 MINUTES**

FREQUENCY:

**EVERY TUESDAY**

**6PM EST**

PURPOSE:

**GENERAL SUPPLEMENT**

**TO LECTURES**

## About Office Hours

DURATION:

**90 MINUTES**

FREQUENCY:

**EVERY SUNDAY**

**10AM EST**

PURPOSE:

**PERSONALIZED HELP**

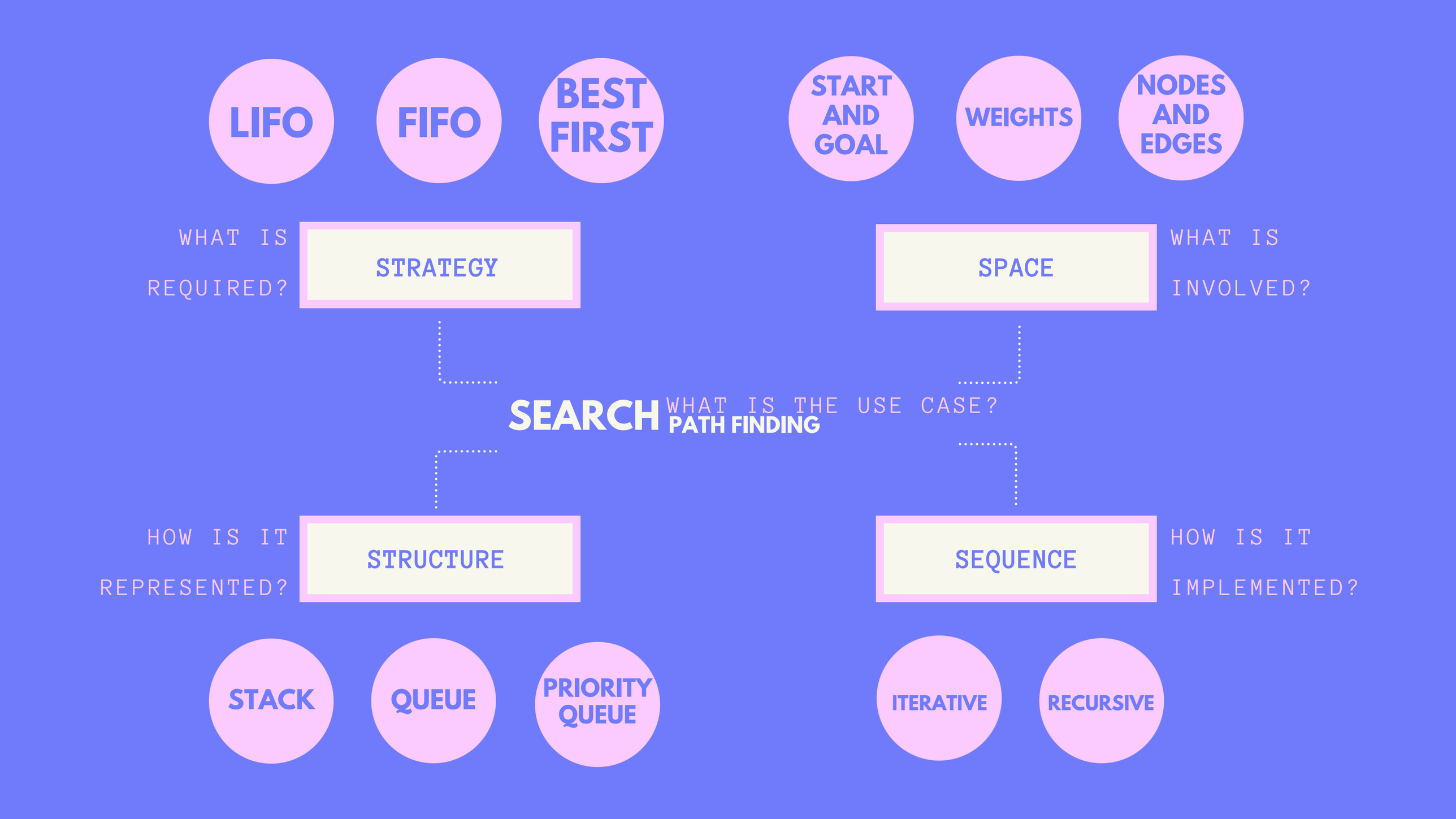
People think that computer science is the art of geniuses, but the actual reality is the opposite, just many people doing things that build on each other, like a wall of mini stones

DONALD KNUTH


**Please help each other!**
**(except for the quizzes)**

# Algorithmic Framework

- WHAT IS INVOLVED?
- WHAT IS REQUIRED?
- HOW IS IT REPRESENTED?
- HOW IS IT IMPLEMENTED?
- WHAT IS THE USE CASE?


LIFO
FIFO
BEST FIRST
START AND GOAL
WEIGHTS
NODES AND EDGES
WHAT IS REQUIRED?
STRATEGY
SPACE
WHAT IS INVOLVED?
SEARCH
WHAT IS THE USE CASE?
PATH FINDING
HOW IS IT REPRESENTED?
STRUCTURE
SEQUENCE
HOW IS IT IMPLEMENTED?
STACK
QUEUE
PRIORITY QUEUE
ITERATIVE
RECURSIVE

# DEPTH FIRST SEARCH

## What is involved?

search space with perfect information

## What is required?

LIFO search strategy

## How is it represented?

a stack data structure

## How is it implemented?

recursively

## What is its use case?

backtracking

# BREADTH FIRST SEARCH

What is involved?

search space with perfect information

What is required?

FIFO search strategy

How is it represented?

a queue data structure

How is it implemented?

iteratively

What is its use case?

shortest path problems

# GREEDY BEST FIRST SEARCH

What is involved?

What is required?

How is it represented?

How is it implemented?

What is its use case?

# A* SEARCH

What is involved?

What is required?

How is it represented?

How is it implemented?

What is its use case?

# DIJKSTRA'S ALGORITHM

What is involved?

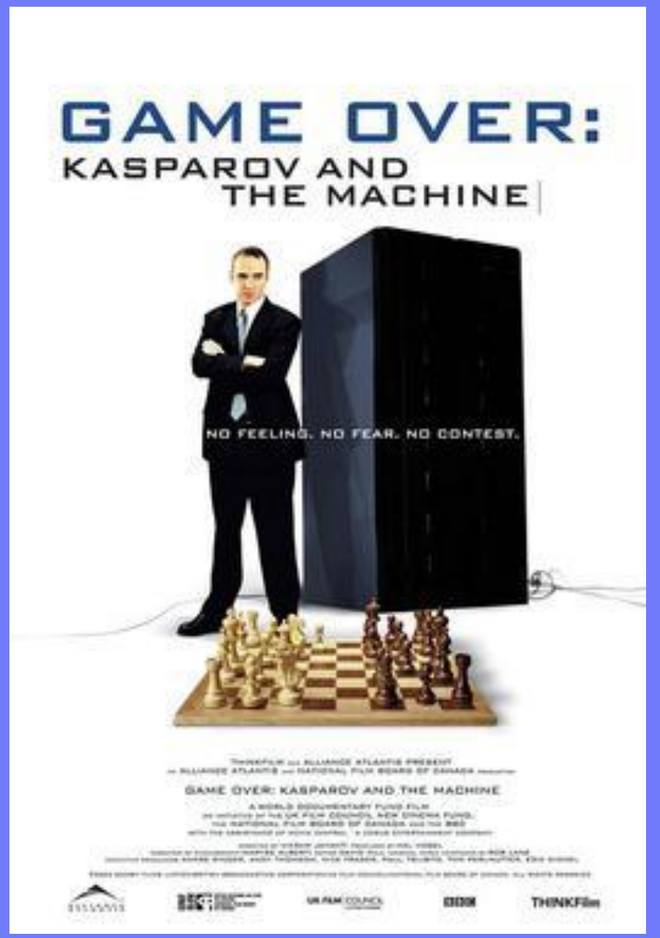
What is required?

How is it represented?

How is it implemented?

What is its use case?

# AI Stories:
# IBM Deep Blue



# Work Environment

Highly recommend coding from
an IDE designed for developers

## PyCharm

## VSCode

# Submitting Projects

## 0. Authorize submit50 at cs50.me

(assuming you already created a GitHub account)

### submit50

```
pip install submit50
```

### Git

```
git remote add origin
https://github.com/[repo]/
[username].git
```

### CS50 IDE

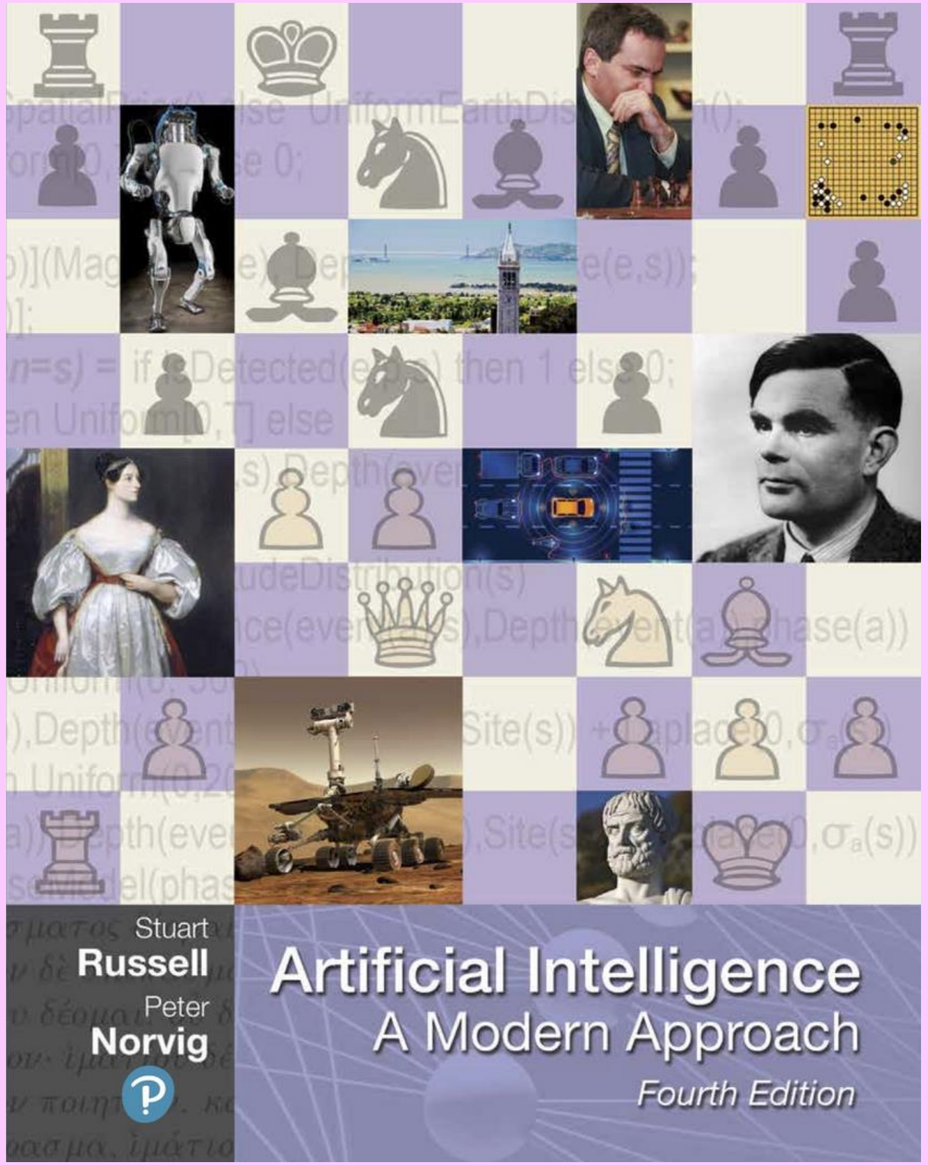
```
sandbox.cs50.io
```

HELPFUL TEXTBOOK

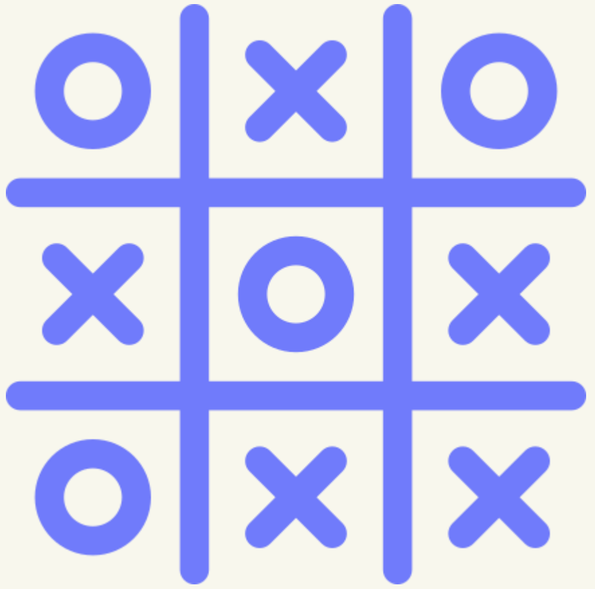
# RUSSELL & NORVIG

## ARTIFICIAL INTELLIGENCE: A MODERN APPROACH

Fourth Edition



# projects



# degrees

## graph theory

A graph G = (V, E) consists of a set of nodes V (or vertices, points) and a set of edges E (links, lines) which illustrate how the nodes in the network are interacting with each other

*definition from Kaggle*

# degrees

```
def shortest_path(source, target):
    """
    Returns the shortest list of (movie_id,
    person_id) pairs that connect the source to the
    target. If no possible path, returns None.
    """
```

**Tip: what should be returned when source == target?**

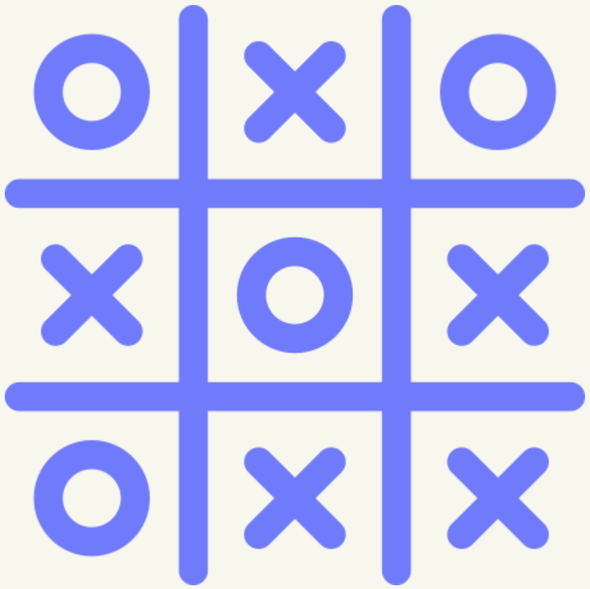
**Stuck? Check out lecture source code maze.py!**

# degrees

## extra challenge

Is searching for the shortest path by starting from the source and ending on the goal the most efficient strategy? How can BFS perform even better, in addition to the project hint?

# tic tac toe



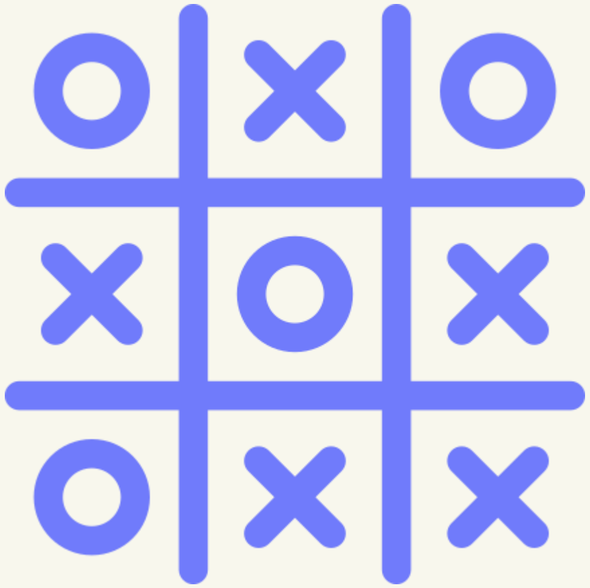
# recursions

```
def factorial(n):
    return 1 if n == 1 else n * factorial(n - 1)
```

# python data structures

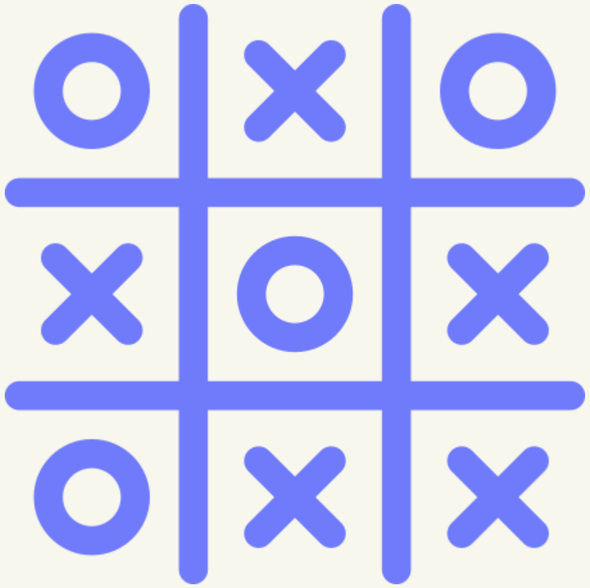

List Tuple Set Dictionary

# tic tac toe



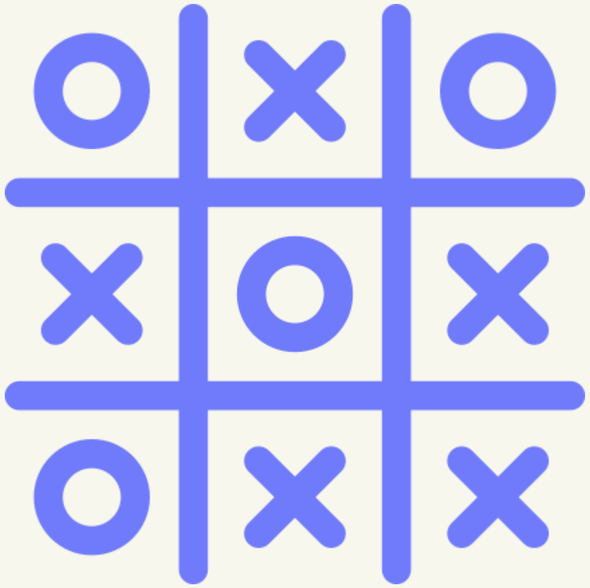
```
def player(board):
    """
    Returns player who has the next turn on a board.
    """
    Tip: think about it as a math problem
```

# tic tac toe



```
def actions(board):
    """
    Returns set of all possible actions (i, j)
    available on the board.
    """
    Tip: be mindful of the python data structure that
    the function should return
```

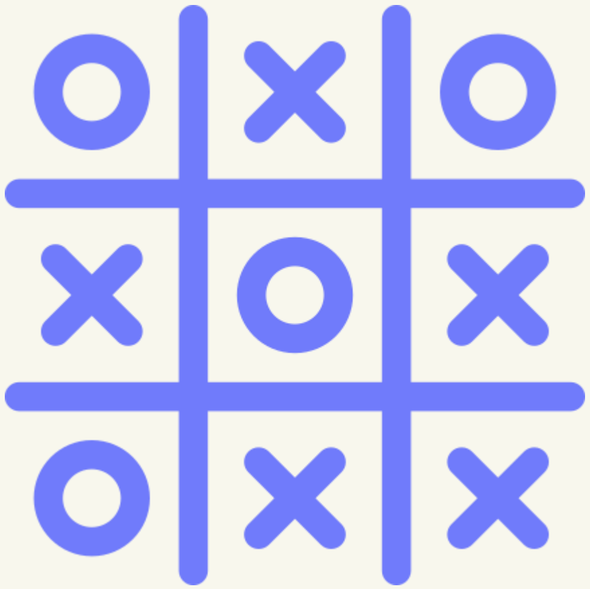
# tic tac toe



```
def result(board, action):
    """
    Returns the board that results from making move
    (i, j) on the board.
    """
```

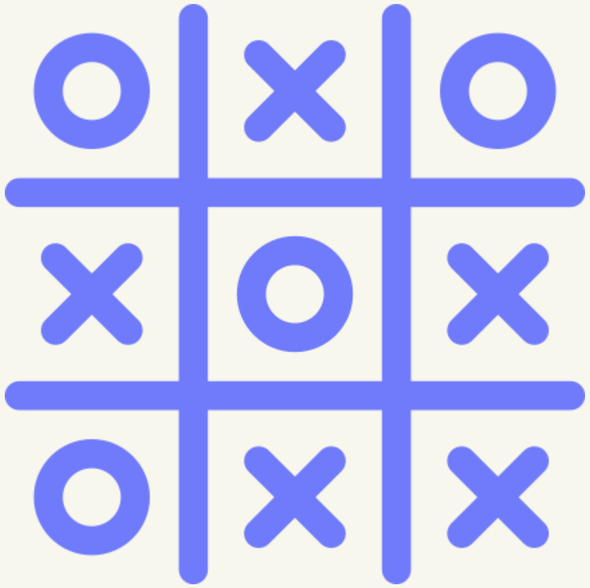
**Tip: what function from the project can help label the board move with the correct player?**

# tic tac toe



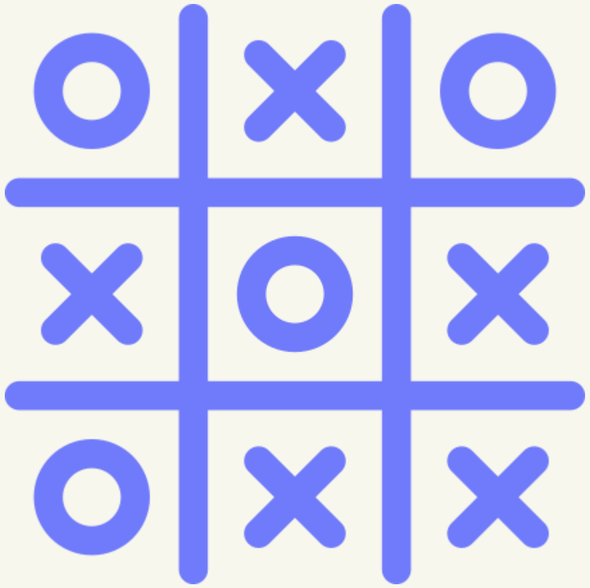
```
def winner(board):
    """
    Returns the winner of the game, if there is one.
    """
    Tip: if your solution is not elegant, that's
    okay!
```

# tic tac toe



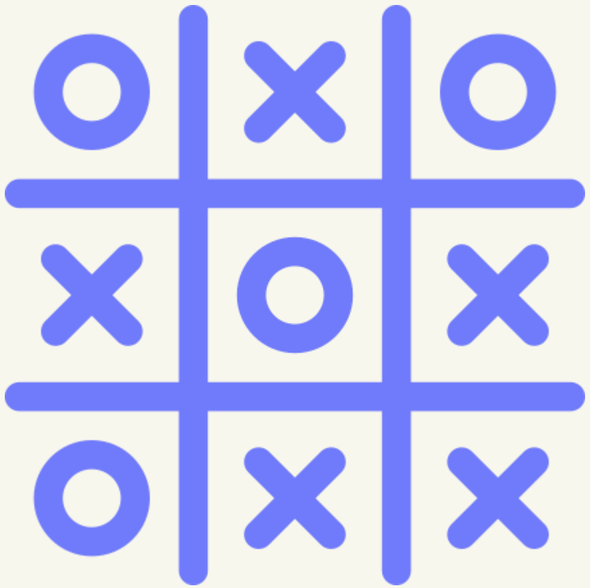
```
def terminal(board):
    """
    Returns True if game is over, False otherwise.
    """
    Tip: how does the program know when there is a
    winner and when all cells have been filled?
```

# tic tac toe



```
def utility(board):
    """
    Returns 1 if X has won the game, -1 if O has won,
    0 otherwise.
    """
    Tip: what function from the project can make
    implementing this really easy?
```

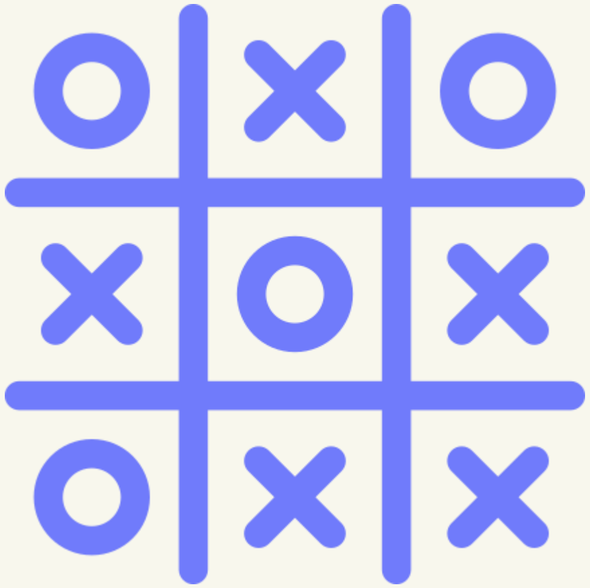
# tic tac toe



## minimax

- max_value helper function
- min_value helper function

# tic tac toe



## unittest tutorial

# these are hard times, but we got this!