# CS 205 AI Project
# Pac-Man Automated Search

## Summary

### Learning Goals/Objective

In this project your goals are to learn how to implement some of the searches we have learned in class in a fun game. You will be implementing Depth First Search, Breadth First Search, A* search, Uniform Cost Search, a greedy search, and various heuristics. You will be able to see first hand what each search is good at and gain a better understanding of where they would be used.

You may work in a group of at most 3 people. All submissions for your group will be group-based. So appoint a person who will be responsible for maintaining the deliverables.

### Due dates

Deliverables due each week, final report due in Week 6.

## Implementation

### Where to begin

Visit the main website for this project (http://ai.berkeley.edu/search.html) and download the main zip file (Berkeley Pac-Man zip). After downloading, unzip the file in the directory of your choice. You can play the game manually by using the command `python pacman.py` from your terminal (if your computer is set up to use Python 3 as default, you must type `python2 pacman.py` instead).

After unzipping you will notice a lot of files, but there are only two main files you will be modifying throughout the entire project: search.py and searchAgents.py. The website has a list of files you might want to look into, and a list of files you can ignore.

### Procedure

01. Week 4 ( Due Oct 28 by 5pm)

a. Run through the tutorial and familiarize yourself with the game
    i. After downloading and playing around with Pac-Man, run through the short tutorial section (the Welcome to Pacman section) on the main website. You will be able to see how the AI can automatically make Pac-Man move and go towards his targets.
b. Implement Questions 1-2
    i. For the first week, you will be implementing DFS and BFS in order to help Pac-Man find a fixed piece of food. All of the search algorithms you will implement after this will be similar to DFS and BFS, so make sure to implement them correctly and the rest will come easily!
    ii. Make sure your code returns a solution for the examples in question 1 and question 2 on the website.
    iii. Run `python autograder.py --q q1` and `python autograder.py --q q2`
c. Implement Question 3-4
    i. Implement UCS and A* Search.
    ii. Make sure your code returns a solution for question 3 and 4 on the website.
    iii. Run `python autograder.py --q q3` and `python autograder.py --q q4`

02. Week 5 (Due Nov 04 by 5pm)
a. Implement Question 5
    i. Use BFS search to find all the corners of the map
    ii. Make sure your code runs for tinyCorners and mediumCorners in question 5 on the website
    iii. Run `python autograder.py --q q5`
b. Implement Question 6
    i. Implement a non-trivial heuristic for your A* search to find the corners of the map
    ii. Make sure your code runs for mediumCorners instructions in question 6 on the website
    iii. Run `python autograder.py --q q6`

03. Week 6 (Due Nov 11 by 5pm)
a. Implement Question 7-8
    i. Use A* search in order to eat all the dots on the map, and implement a suboptimal greedy search to eat all the dots
    ii. Make sure your code runs for the searches in questions 7 and 8 on the website

iii. Run `python autograder.py --q q7` and `python autograder.py --q q8`

b. Write the final report
   i. In your report, describe.
   ii. Your final report must be a **maximum** of 2 pages. Any more than 2 pages and you will lose points!

## Deliverables

Upload deliverables via Google Docs. You will be graded on a mix of three things: the README file, if your code runs on all the instructions for the questions, and the autograder grade for that question. Think of your README as a research log — nice documentation of your work and efforts, but in a way that will be easy for us to scan and interpret. That is, make it concise, informative, detailed, and organized. Keep it concise, which means short yet dense and informative.

Follow these instructions **carefully!** You will lose points if your Google Drive does not have the correct format:

- Create a Google Drive folder for PacMan.
- Add a README with **only** your team members' names in the **root** folder.
- Make the PacMan folder viewable for anyone with an R'Mail and the Link.
- Create 3 sub-folders in PacMan called "Week 4" ... up to "Week 6."
- Create a thread in the slack with all your teammates and the TA (/readers), this is where you will ask all your questions on the project. Send a link to the TA (/readers) of your main Google Drive folder.
- Upload the necessary files below in the sub-folder.
  - search.py
  - searchAgents.py
- Summarize your progress and learnings in the README in a paragraph or two.
- Include a file listing and description for each of the other files in the folder in the README.
- Add screenshots as appropriate (nicely, don't make the file too long).
- Make sure to submit your files to Google Drive before 5pm. Fill-out the Google Form (to be provided on Slack) to submit and timestamp your work.

  UPDATE: Timestamping Google Form can be found here.

1. Due Week 4

    a. Upload search.py (with your DFS and BFS code implemented)
    b. Create/upload screenshots of your successful runs on the commands on the website for questions 1 and 2, add to README
    c. Answer the questions in "Question 1" (label it clearly as such) in your README.
    d. Upload search.py (with your UCS and A* code implemented)
    e. Upload screenshots of your successful runs on the commands on the website for questions 3 and 4, add to README

3. Due Week 5

    a. Upload searchAgents.py with the CornersProblem implemented
    b. Upload screenshots of your successful runs on the commands on the website in question 5, add to README
    c. Upload searchAgents.py with your cornersHeuristic implemented
    d. Upload screenshots of your successful runs on the commands on the website in question 6, add to README

5. Due Week 6

    a. Upload search.py file.
    b. Upload searchAgents.py with your foodHeuristic and findPathToClosestDot implemented
    c. Upload screenshots of your successful runs on the commands on the website in questions 7 and 8, add to README
    d. Upload your **maximum** 2 page final report in the ROOT ("PacMan") folder.

## Materials & Resources

The project will be run in Python 2. If you choose to run it with Python 3 you are responsible yourself for changing the project files to make it work with Python 3.

a. Project Materials:
    a. Main Project Website: http://ai.berkeley.edu/search.html
b. Resources:
    a. Slides (in Slack)

b. Book (AI a Modern Approach)
c. Python 2 documentation: https://docs.python.org/2.7

## Assessment

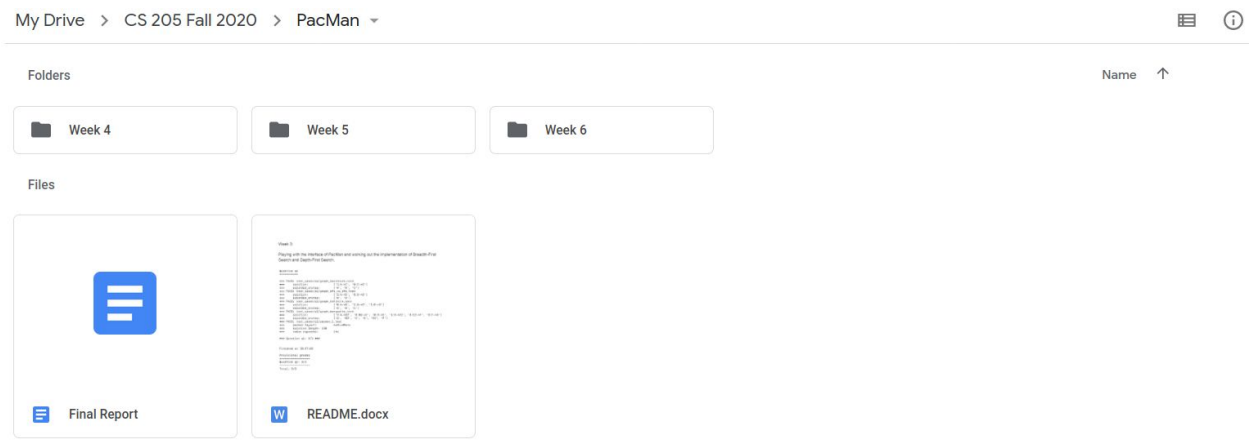
You are responsible for uploading the required materials to the Google Drive folder.

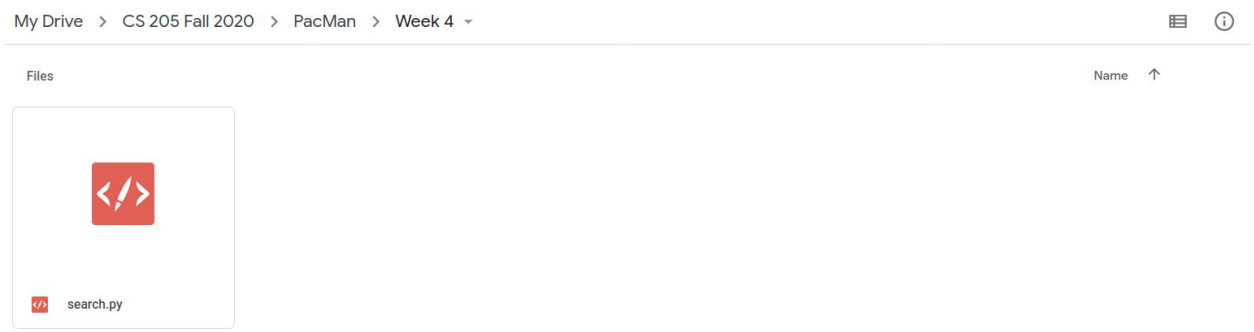
**The Final Report**: Summarize in a single report all you have accomplished and learned as a team. Highlight in the report the activities you found most challenging and why, the activities you found most interesting and why, or the activities you simply hated and why. Tell us what you thought! Discuss the team dynamic, where there challenges you had to overcome? Take a moment to describe what you are most proud of accomplishing (and why). Show off to us! You only have two pages, so use them wisely. We don't want to read fluff and platitudes or pandering. We want a serious analysis and debriefing of your project work.

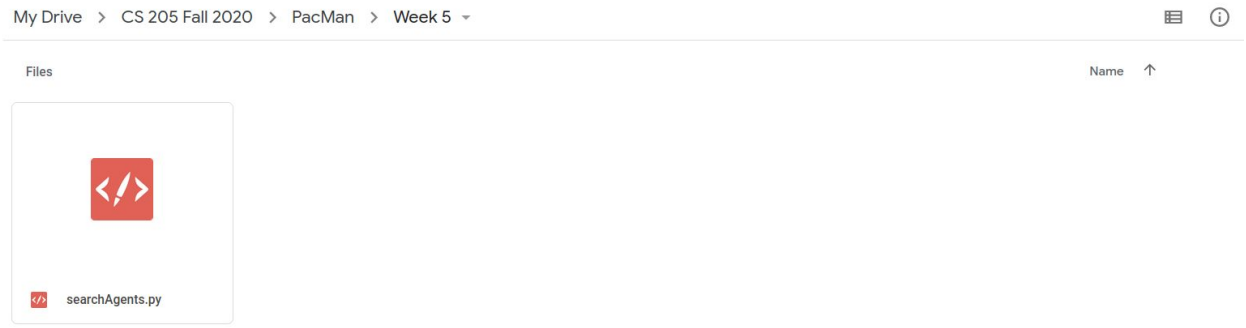
Not sure how to write a good project report? Google it! Ten times. You can figure it out.

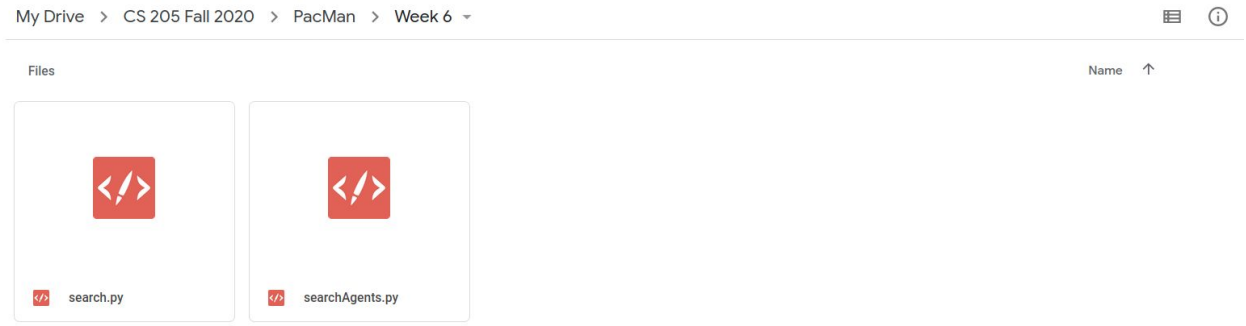
**Extra points for creativity.** For example: Maybe you want to make a video? Maybe you want to create a newspaper print, showcasing your work. Maybe you create a new game that makes us search for the answers to your project. Who knows! Impress us. Extra points for creativity.

What your Google Drive should look like:




My Drive > CS 205 Fall 2020 > PacMan > Week 4
Files
Name
search.py


My Drive > CS 205 Fall 2020 > PacMan > Week 5
Files
Name
searchAgents.py


My Drive > CS 205 Fall 2020 > PacMan > Week 6
Files
Name
search.py
searchAgents.py