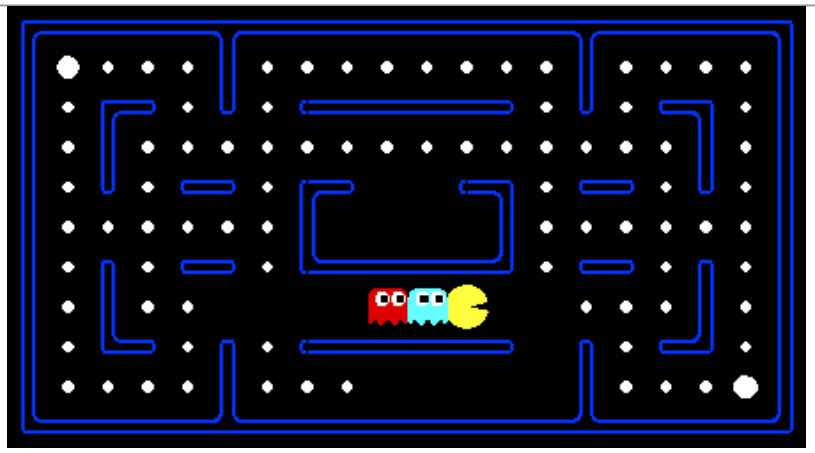
p2_multiagent_introduction
Project 2: Multi-Agent Pacman



Pacman, now with ghosts.
Minimax, Expectimax,
Evaluation.

Introduction

In this project, you will design agents for the classic version of Pacman, including ghosts. Along the way, you will implement both minimax and expectimax search and try your hand at evaluation function design.

The code base has not changed much from the previous project, but please start with a fresh installation, rather than intermingling files from project 1.

As in project 1, this project includes an autograder for you to grade your answers on your machine. This can be run on all questions with the command:

```
python autograder.py
```

It can be run for one particular question, such as q2, by:

```
python autograder.py -q q2
```

It can be run for one particular test by commands of the form:

```
python autograder.py -t test_cases/q2/0-small-tree
```

By default, the autograder displays graphics with the `-t` option, but doesn't with the `-q` option. You can force graphics by using the `--graphics` flag, or force no graphics by using the `--no-graphics` flag.

See the autograder tutorial in Project 0 for more information about using the autograder.

The code for this project contains the following files, available as a zip archive.

Files you'll edit:	
`multiAgents.py`	Where all of your multi-agent search agents will reside.
Files you should read but NOT edit:	
`pacman.py`	The main file that runs Pacman games. This file also describes a Pacman `GameState` type, which you will use extensively in this project
`game.py`	The logic behind how the Pacman world works. This file describes several supporting types like AgentState, Agent, Direction, and Grid.
`util.py`	Useful data structures for implementing search algorithms.
Files you can ignore:	

graphicsDisplay.py	Graphics for Pacman
graphicsUtils.py	Support for Pacman graphics
textDisplay.py	ASCII graphics for Pacman
ghostAgents.py	Agents to control ghosts
keyboardAgents.py	Keyboard interfaces to control Pacman
layout.py	Code for reading layout files and storing their contents
autograder.py	Project autograder
testParser.py	Parses autograder test and solution files
testClasses.py	General autograding test classes
test_cases/	Directory containing the test cases for each question
multiagentTestClasses.py	Project 2 specific autograding test classes

Files to Edit and Submit: You will fill in portions of `multiAgents.py` during the assignment. You should submit this file with your code and comments. Please *do not* change the other files in this distribution or submit any of our original files other than this file.

Evaluation: Your code will be autograded for technical correctness. Please *do not* change the names of any provided functions or classes within the code, or you will wreak havoc on the autograder.

Getting Help: You are not alone! If you find yourself stuck on something, take advantage of our piazza discussion forum.

Discussion: Please be careful not to post spoilers.