Artificial Intelligence Assignment #3

Due date: Friday, 19, May 2017, 11:55 pm
No Late Submission

Programming Language:
- Python >= 3.5
- There is no external library required for this assignment.
- Possible to use any IDE.

Grading:
- Total: 20 points
- Ranking: 6 points
- QLearningAgent_autograder: 5 points (1 question)
- ApproximateQAgent_autograder: 9 points (3 questions)

Files to Edit and Submit:
- **qlearningAgents.py**: Q-learning agents for Pacman
- **featureExtractors.py**: Classes for extracting features on (state, action) pairs. Used for the approximate Q-learning agent (in qlearningAgents.py)

Useful Files (do not modify):
- learningAgents.py: Defines the base classes QLearningAgent, which your agents will extend.
- util.py: Utilities, including util.Counter, which is particularly useful for Q-learners.
- pacman.py: Contains logic for pacman game.

Given Files (do not modify):
- environment.py: Abstract class for general reinforcement learning environments. Used by gridworld.py
- game.py: Configuration settings for the game.
- ghostAgent.py: Manages ghost agent type.
- layout.py: Manages the game board.
- graphicsUtils.py: Graphics utilities.
- 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.
- reinforcementTestClasses.py: Assignment 3 specific autograding test classes.
- Other files: Do not worry about other files.

Need to Do:
- Add your univ_id (University number) and Password (phone number in YSCEC) at the top part of qlearningAgents.py file to submit your score.
- Need to implement “qlearningAgents.py”. (QLearningAgent class and ApproximateQAgent class)
- Remove “util.raiseNotDefined()” for each function you are implementing.
- To use enhanced features for ApproximateQAgent, add your own features in CustomExtractor class in “featureExtractors.py”. Feel free to add any functions for

your CustomExtractor class.
- Submit your average score for “ApproximateQAgent”.

Score Submission:
- Only able to submit your average score in this condition. (numTraining=50, numGames=60, agent=ApproximateQAgent, and layout=mediumClassic)
- You will see “Do you want to submit your average score? (y/n):” message in the console. If you want to submit, please type “y” in the command line. Otherwise, type “n”.
- You can submit your score multiple times in every 10 minutes.

Extra Information:
- Quiet mode is available. (Only receive the scores without any visualization). (Example: python pacman.py -q {your_options})
- More information: python pacman.py –help
- You can find more information in http://ai.berkeley.edu/reinforcement.html

Pacman Score:
- Eat Food: + 10
- Each Action: - 1
- Eat Ghost: + 200
- Win: + 500
- Lose: - 500

How to Test (Command):
- QLearningAgent: python pacman.py -p PacmanQAgent -x 2000 -n 2010 -l smallGrid
  Use layout=smallGrid, trainNum=2000, totalRun=2010
- ApproximateQAgent: python pacman.py -p ApproximateQAgent -a extractor=SimpleExtractor -x 50 -n 60 -l mediumClassic
  Use layout=mediumClassic, trainNum=50, totalRun=60, and use SimpleExtractor (default feature extractor provided). You may change to a different extractor in featureExtractors.py

How to Test Using Autograder (Command):
- Both: python autograder.py
- QLearningAgent: python autograder.py -q q1
- ApproximateQAgent: python autograder.py -q q2

Submission:
- Submit your file in YSCEC → Assignment #3.
- Create a UniversityNumber.zip (2016000000.zip) file that contains only featureExtractors.py and qlearningAgents.py

How to Test Using Autograder (Command):
- Both: python autograder.py
- QLearningAgent: python autograder.py -q q1
- ApproximateQAgent: python autograder.py -q q2