



Bookmarks



Bookmark

▶ Week 1

▶ Week 2

▶ Week 3

▶ Week 4

▶ Week 5


▼ Week 6


Lecture 10:  
Reinforcement  
Learning (edited)

Lecture 10:  
Reinforcement  
Learning (live)

Lecture 11:  
Reinforcement  
Learning II (edited)

Lecture 11:  
Reinforcement  
Learning II (live)

Homework 5:  
Reinforcement  
Learning  
Homework 

Project 3:  
Reinforcement  
Learning  
Project 3 

Midterm 1  
Preparation

Week 6 &gt; Project 3: Reinforcement Learning &gt; p3\_rl\_q5\_epsilon\_greedy

## Question 5 (3 points): Epsilon Greedy

Complete your Q-learning agent by implementing epsilon-greedy action selection in `getAction`, meaning it chooses random actions an epsilon fraction of the time, and follows its current best Q-values otherwise. Note that choosing a random action may result in choosing the best action - that is, you should not choose a random sub-optimal action, but rather *any* random legal action.

```
python gridworld.py -a q -k 100
```

Your final Q-values should resemble those of your value iteration agent, especially along well-traveled paths. However, your average returns will be lower than the Q-values predict because of the random actions and the initial learning phase.

You can choose an element from a list uniformly at random by calling the `random.choice` function. You can simulate a binary variable with probability `p` of success by using `util.flipCoin(p)`, which returns `True` with probability `p` and `False` with probability `1-p`.

To test your implementation, run the autograder:

```
python autograder.py -q q5
```

With no additional code, you should now be able to run a Q-learning crawler robot:

```
python crawler.py
```

If this doesn't work, you've probably written some code too specific to the `GridWorld` problem and you should make it more general to all MDPs.

This will invoke the crawling robot from class using your Q-learner. Play around with the various learning parameters to see how they affect the agent's policies and actions. Note that the step delay is a parameter of the

- ▶ Week 7
- ▶ Week 8
- ▶ Week 9
- ▶ Week 10
- ▶ Week 11
- ▶ Week 12
- ▶ Week 13
- ▶ Week 14

simulation, whereas the learning rate and epsilon are parameters of your learning algorithm, and the discount factor is a property of the environment.

© All Rights Reserved



© edX Inc. All rights reserved except where noted. EdX, Open edX and the edX and Open EdX logos are registered trademarks or trademarks of edX Inc.

POWERED BY  
OPENedX

