Single dot with static monster Reinforcement Learning Assignment

In this assignment, we want to make Pacman reach the dot and eat it. We have two programs: singleDotProblemStaticMonster.py and singleDotStaticMonster.py.

In singleDotProblemStaticMonster.py, we have four classes:

1. Monster class:
   1. The constructor \_\_init\_\_ function stores the position and color of the monster.
   2. display method to display the color and position of the monster.
   3. \_\_eq\_\_ method which is a special method that is used to define how objects of a class are compared for equality.
   4. \_\_hash\_\_ method which is a special method that is used to define how an object is hashed.
2. Agent class:
   1. The constructor \_\_init\_\_ that store the position of the agent
   2. Display method to display the agent position
   3. \_\_eq\_\_ method which is a special method that is used to define how objects of a class are compared for equality.
   4. \_\_hash\_\_ method which is a special method that is used to define how an object is hashed.
3. **State class** where it stores the agent position (that is Pacman).
   1. The constructor \_\_init\_\_ function to store the agent class.
   2. Define special methods:
      1. Define \_\_eq\_\_ method which is a special method that is used to define how objects of a class are compared for equality. It is used to define how instances of a class are compared for equality using the == operator. When you use == to compare two objects, Python will call the \_\_eq\_\_ method to determine if the two objects are considered equal.
      2. Define \_\_hash\_\_ method which is a special method that is used to define how an object is hashed.

These two methods are part of the object’s “magic methods” which allow you to override default behavior in Python. It is used to define how instances of a class are hashed, which is important when objects are used in data structures like sets or as dictionary keys. If you define \_\_eq\_\_ in your class, you should also define \_\_hash\_\_ to maintain consistency between equality and hash values. The \_\_hash\_\_ method returns an integer that represents the hash value of the object. The rule is that if two objects are considered equal (\_\_eq\_\_ returns True), they must have the same hash value.

1. **Problem Class:** Where it stores the following methods:
   1. readMaze: to read the maze stored in a file name, this method defines some variables like wall positions, agent position, dots positions, potential\_moves, xMax (maximum width of maze) and yMax (maximum height of maze).
   2. isWall: returns true if the position is in the wall.
   3. getStartState: returns the start state of the problem that is the start agent position.
   4. isValidMove: that returns true if the agent position is valid, i.e. not inside the wall or outside maze boundaries.
   5. isTermina: returns true if the state is a terminal state, terminal state is when agent position is at the dot position.
   6. Reward: returns the reward for every state, at terminal state it returns a reward of 10 otherwise it returns a reward of 0.
   7. Transition: returns a list of valid new state and action, the format of the list is (state, action)

The file singleDotStaticMonster.py is the learning agent program where it uses the value iteration algorithm. Your assignment is to complete the code that is missing. Use the singleDot.py as a guide.

The final output should be:

A screenshot of a game

Description automatically generated

Make sure your code runs with no errors and produce the correct output. Create another github repository and save all files in the github repository.

Submit the link to your github repository in Canvas.