############## Pacman Competition Instructions ##############

Competition:

Implement your own algorithm in the class **CompetitionAgent** within the **pacmanAgents.py** file. The goal is to get the highest score among all the competitors.

Notes:

- Python 2.7 is required to run the Framework.
- All your code must be inside the **pacmanAgents.py** file.
- Only submit **pacmanAgents.py** file and 1 page summary (maximum 500 words) that explains the idea behind your algorithm. If you submit anything else you will be disqualified.
- You will be **disqualified** if:
  - you try to change any of the system params or use another forward model.
  - each move takes more than 1 second.
  - you use any external library (Python standard library (https://docs.python.org/2.7/library/) and Numpy are allowed).
  - All your code is not inside the **pacmanAgents.py** file.
  - you submit more than 1 page summary.
- You are only allowed to use these system functions (accessing/changing any other functions or variables is considered cheating):
  - **state.getLegalPacmanActions():** returns all the legal actions in this state (actions that make PacMan moves).
  - **state.getAllPossibleActions():** returns all the possible actions (Directions.North, Directions.South, Directions.East, Directions.West).
  - **state.generatePacmanSuccessor(action):** returns the next state if pacman take a certain action (return a new copy, doesn't modify the current state).

  - **state.isWin():** checks if this state is a win state.
  - **state.isLose():** checks if this state is a lose state.
  - **state.getPacmanPosition():** returns the current position of pacman.
  - **state.getGhostPositions():** returns an array of the positions of all the ghosts.
  - **state.getScore():** returns the current score.
  - **state.getPellets():** returns an array of the position of all the pellets.
  - **state.getCapsules():** returns an array of the position of all power pellets.
  - **state.getWalls():** return a 2D boolean array of all the level layout.
- The forward model (**generatePacmanSuccessor**) is limited to a certain amount of calls, don't waste them. If you exceed the limit, **None** will be returned.

How to run:

- To play pacman: `python pacman.py`
- To run a certain agent using graphics use the following command: `python pacman.py -p AgentName`

This competition is voluntary, and only submissions that either perform very well or contain interesting ideas will receive any score (all score from this competition count as a bonus). It is possible to achieve full marks in the course without participating in the competition - only participate in the competition if you think you have a good idea you want to try out. Group work is allowed but the points will be split equally between team members.

Your submission will be scored by Julian and the TAs based on its performance, and on the quality of the code and writeup. Submissions that simply reuse existing assignment solutions without any interesting additions, and which do not perform very well, will not receive any score.