In search.py, you will implement generic search algorithms which are called by Pacman agents (in searchAgents.py).
from game import Directions
s = Directions.SOUTH
w = Directions.WEST
return [s, s, w, s, w, w, s, w]

def depthFirstSearch(problem):
    ""
    Search the deepest nodes in the search tree first [p 85].
   "
    Your search algorithm needs to return a list of actions that reaches
    the goal. Make sure to implement a graph search algorithm [Fig. 3.7].
    To get started, you might want to try some of these simple commands to
    understand the search problem that is being passed in:
    print "Start:", problem.getStartState()
    print "Is the start a goal?", problem.isGoalState(problem.getStartState())
    print "Start’s successors:", problem.getSuccessors(problem.getStartState())
    ""
    *** YOUR CODE HERE ***
    util.raiseNotDefined()

def breadthFirstSearch(problem):
    """Search the shallowest nodes in the search tree first. [p 81]"
    *** YOUR CODE HERE ""
    util.raiseNotDefined()

def uniformCostSearch(problem):
    """Search the node of least total cost first."
    *** YOUR CODE HERE ""
    util.raiseNotDefined()

def nullHeuristic(state, problem=None):
    """
    A heuristic function estimates the cost from the current state to the nearest
    goal in the provided SearchProblem. This heuristic is trivial.
    """
    return 0

def aStarSearch(problem, heuristic=nullHeuristic):
    """Search the node that has the lowest combined cost and heuristic first."
    *** YOUR CODE HERE ""
    util.raiseNotDefined()