5			(3,5)	(4,5)	(5,5) Start
4			(3,4)	(4,4)	(5,4)
3				(4,3)	(5,3)
2					
1	(1,1) goal				
	1	2	3	4	5

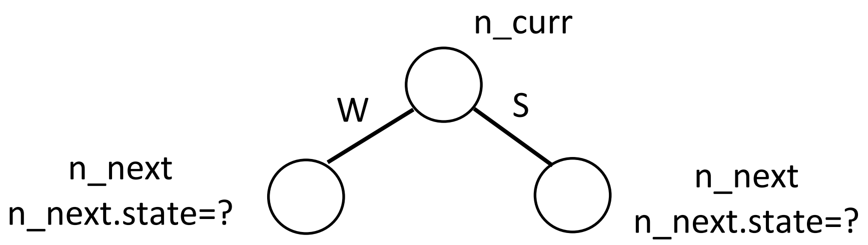
5x5 grid: use the search algorithm to find a path to go from the start location, e.g., (5,5) to the destination
There may be walls blocking the way of Pacman

States: (x,y) location
Actions: go North, South, West, East;

A*: heuristic function: Manhattan heuristic
Two locations: $(x_1, y_1)$and (x_2, y_2)
The Manhattan distance of these two locations are:
$|x_1 - x_2| + |y_1 - y_2|$

f = g + h

Construct a Search Tree: nodes



n_curr is an object of Class Node
n_next is an object of Class Node

four attributes:
n_curr.state: (x,y)
n_curr.parentnode: another object of Class Node
n_curr.action
n_curr.path_cost

e.g. In the 5x5 maze, if n_curr is the root node, then n_curr.state = (5,5)
If n_curr is the goal node, then n_curr.state = (1,1)

Before visiting/expanding a node (meaning its successors will be generated and will enter the frontier queue), first check whether node.state is already visited? i.e. is that location (x,y) already visited?

If yes, do not expand this node.
If no, then run a goal test.

- If it's the goal node, then return the solution (a sequence of actions)
- If it's not the goal node, then expand this node. That is, this node's successors are generated one by one, and enters the Frontier queue (BFS: Queue; A*: priority Queue)

Class Queue and Class Priority Queue: defined in util.py

In addition to "search.py", you may need to have some self-defined functions, such as getSolution

searchAgents.py:

```
class PositionSearchProblem(search.SearchProblem):
  # functions that you may use:
  def getStartState(self):
  def goalTest(self, state):
  def getActions(self, state):
  def getResult(self, state, action):
  def getCost(self, state, action): # step cost, or incremental cost, not
g(n)
```

The counter for expanded nodes: problem._expanded

```
searchAgents.py
->class PositionSearchProblem(search.SearchProblem):
    ->def getActions(self, state):
        ->self._expanded += 1 # DO NOT CHANGE
```

```
# also in searchAgents.py
def manhattanHeuristic(position, problem, info={}):
```

Debug in Anaconda: start from pacman.py

Run->Configuration per file->General settings->Command line options:

```
-l smallMaze -p SearchAgent -a fn=breadthFirstSearch
```