COEN 166 Artificial Intelligence

Lab Assignment #4: Search II

Name: Tanner Kaczmarek   ID: 00001443463

Problem 1 A* Search

Function 1:

```
def aStarSearch(problem, heuristic=nullHeuristic):
    """Search the node that has the lowest combined cost and heuristic first."""
    "*** YOUR CODE HERE ***"
    from game import Directions
    node = Node(problem.getStartState(), None, None, 0)
    frontier = util.PriorityQueue()
    frontier.push(node, heuristic(problem.getStartState(), problem))
    found = False
    #visited = []

    while(found != True):
        if frontier.isEmpty():
            return
        node = frontier.pop()
        if problem.goalTest(node.state):
            found = True
            continue
        #visited.insert(0, node.state)
        for acts in problem.getActions(node.state):
            child_state = problem.getResult(node.state, acts)
            child_cost = problem.getCost(node.state, acts)
            child_node = Node(child_state, node, acts, child_cost+node.path_cost)
            #if (acts in visited):
                #continue
            frontier.update(child_node, (heuristic(child_state, problem)+child_node.path_cost))

    actions = []
    while not node.state == problem.getStartState():
        actions.insert(0, node.action)
        node = node.parent
    return actions
```

Comment: The solution for my A* Search only has one function. The first block of code from importing directions to “found = false” is just creating variables I will need to use to find my solution. I used PriorityQueue to find my solution which is special kind of queue. A regular queue is a FIFO data structure while priority queue is first in and then highest priority first out data structure.   When you update another node into a priority queue it will need its priority. The priority will be the A* search function of the path cost to the node + the result of the heuristic function. Whatever number is lowest has the highest priority.

The next part of my code is the while loop which will go until the current node's state passes the goal test. For each iteration of the while loop it will pop off the frontier priority queue, the highest priority node. The node then is ran through all its possible actions and is updated into the frontier. The child node's cost is the total cost of the path to get to the current location. The priority added to the queue is the heuristic function result of the child node's state plus the node path cost.

The last part of my A* search gets all the actions of the function into a list called actions. It starts with the goalNode from the while loop and iterates through its parent until the current node's state is equal to the startState. While it goes through, it adds the action of the node to the actions list.