COSE361(03) Artificial Intelligence

Assignment #1

2022320033 박종혁

a. Capture the result of autograder.py in terminal.

```
Finished at 11:27:50

Provisional grades
==================
Question q1: 3/3
Question q2: 3/3
Question q3: 3/3
Question q4: 3/3
Question q5: 0/3
Question q6: 0/3
Question q7: 0/4
Question q8: 0/3
------------------
Total: 12/25
```

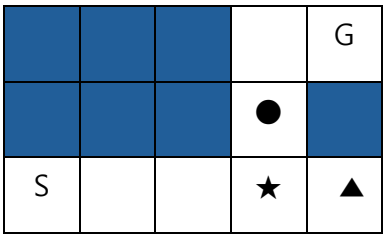
b. Three discussions on three different algorithms (DFS, BFS, A*) when playing Pacman.

- In my implementation, DFS explored 146 nodes and found the path with cost of 130. On the other hand, BFS explored 269 nodes and found the path with cost of 68 in mediumMaze search. It seems that DFS explores less nodes than BFS does, but it is a lucky case, not always. DFS continues until there are no more nodes to explore, so once it goes the wrong way it wastes a lot of time. In addition, unlike BFS, DFS does not guarantee the shortest path. Therefore, I think BFS is more efficient one as a maze search algorithm than DFS in that it guarantees an average search time and the shortest path.

- Another famous heuristic function is 'Euclidean distance',

$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

In case of the following maze layout, the Euclidean heuristic might be more effective than the Manhattan heuristic:



- S: Start
- G: Goal
- Blue: Wall

When current state is ★, the Manhattan heuristic will measure f values of ● and ▲ are the same. On the other hand, the Euclidean heuristic will measure f value of ● is smaller than that of ▲. The real optimal solution is ●, so the Euclidean heuristic is more effective than Manhattan heuristic at least in this example.

If we consider the search problem where Pacman could move in diagonal directions, we can say that the Euclidean heuristic is more effective than the Manhattan heuristic in most cases. It's because Manhattan heuristic does not consider diagonal move at all, but Euclidean does.

- Q: Are there cases A* is slower than UCS?

A: Yes. I'll explain with Manhattan heuristic. In A* search, the heuristic function h does not postulate the existence of walls, which means the algorithm tends to measure the cost of a path blocked by walls less than its actual cost. That is, A* with Manhattan heuristic may be slower than naïve UCS if the path is quite meandering. There is good example YouTube video showing such problem: https://youtu.be/g024lzsknDo?feature=shared&t=85

c. I implemented myHeuristic using Chebyshev distance,

$$\max(|x_1 - x_2|, |y_1 - y_2|)$$

Chebyshev heuristic is always smaller than Manhattan distance, therefore Chebyshev heuristic is admissible too, which means it finds out the shortest path.

```
python pacman.py -l mediumMaze -p SearchAgent -a fn=astar,heuristic=myHeuristic
```

```
[SearchAgent] using function astar and heuristic myHeuristic
[SearchAgent] using problem type PositionSearchProblem
Path found with total cost of 68 in 0.0 seconds
Search nodes expanded: 228
Pacman emerges victorious! Score: 442
Average Score: 442.0
Scores:        442.0
Win Rate:      1/1 (1.00)
Record:        Win
```

It resulted in 228 nodes expanded. Comparing to Manhattan (221), Euclidean (226) and null (269), it seems helpful to finding path to the Goal in mediumMaze. The result in the bigMaze was in the same order. Since it uses either x difference or y difference only, I think, it has less effect than Manhattan and Euclidean in most cases.