Homework 1 - Report

- Explain how you decide on the heuristics you determine.

  I had to decide between speed and accuracy. In this homework, we are dealing with mazes that are not so huge. For me, accuracy is the most important thing. So, it needs to have high accuracy. Even if we have something big, today's computers have quite good systems. They can deal with more operation. Manhattan distance is the most common heuristic function for mazes. As I said I care accuracy, Manhattan distance place less emphasis on outliers. It reduces all the wrong paths. Also, on the Internet, people suggest this method to solve problems.

- Can the agent find all possible paths from the starting tile to the goal tile? How can this be made possible?

  a) BFS doesn't keep all paths. However, we can easily modify the algorithm to store a list of possible predecessors rather than one predecessor.
  b) DFS is also like BFS.
  c) UCS with costs can't find all the paths. It traverses with the help of cost on the tile.
  d) UCS without costs can find all the paths. UCS turns to BFS without costs.
  e) Admissable A* Search can't find all the possible paths. It traverses with a heuristic. Heuristic reduces the paths.
  f) Inadmissable A* Search also can't find all the paths. Overestimating heuristic only makes algorithm solve the problem faster.

- Can the agent find the path from the starting tile to the goal tile with minimum number of actions? If so, how? If not, what are the causes?

  a) BFS can find the minimum action path. Because it starts at the tree root, and explores all of the neighbor nodes at the present depth prior to moving on to the nodes at the next depth level. Also, it doesn't care about costs on the tile.
  b) DFS can find the minimum action path. It also doesn't care about costs on the tile. The algorithm starts at the root node and explores as far as possible along each branch before backtracking. It just goes all the way down. However, if the graph is infinite, DFS may not find a solution.
  c) UCS with costs can't find the minimum action path. Because it cares about costs on the tile. With the help of priorities, if it finds a tile that can be reached with a less path cost, then it can change its path. So, in my maze, the minimum action is not the one with minimum cost.
  d) UCS without cost can find the minimum action path. It turns to BFS without costs.
  e) Admissable A* Search can't find the minimum action path. Because it cares about costs on the tile and uses a heuristic function to reach the goal. It does this by maintaining a tree of paths originating at the start node and extending those paths one edge at a time until its termination criterion is satisfied. It does so based on the cost of the path and an estimate of the cost required to extend the path all the way to the goal.
  f) Inadmissable A* Search can't find the minimum action path. With the overestimated heuristic, it finds worse rather than admissible one.

- Can the agent find the path that it reaches the goal tile with the minimum cost in movement points? If so, how? If not, what are the causes?

a) BFS can't find the minimum cost path. Because it doesn't care about costs on the tile. It starts at the tree root, and explores all of the neighbor nodes at the present depth prior to moving on to the nodes at the next depth level.
b) DFS can't find the minimum cost path. Because it also doesn't care about costs on the tile. The algorithm starts at the root node and explores as far as possible along each branch before backtracking.
c) UCS with costs can find the minimum cost path. Because it cares about costs on the tile. With the help of priorities, if it finds a tile that can be reached with a less path cost, then it can change its path.
d) UCS without costs can't find the minimum cost path. BFS can't find the minimum one and UCS turns to BFS and then, UCS without costs also can't find.
e) Admissable A* Search can find the minimum cost path. Because it cares about costs on the tile and uses a heuristic function to reach the goal. It does this by maintaining a tree of paths originating at the start node and extending those paths one edge at a time until its termination criterion is satisfied. It does so based on the cost of the path and an estimate of the cost required to extend the path all the way to the goal.
f) Inadmissable A* Search can't find the minimum one. It tries to get to the goal faster but it loses accuracy.

- What would happen if additional movement points were negative?

a) BFS won't get affected. Because it only focuses the path, not the cost.
b) DFS also won't be affected. The same reason with the BFS is valid.
c) UCS with costs will get affected because meeting with a negative one changes all the order of nodes. It can make nonsense decisions.
d) UCS without cost will be the same with BFS.
e) Admissable A* Search will get affected. Same reasons as UCS with costs. Heuristic will also get affected. Even it may not be able to solve the problem.
f) Inadmissable A* Search will get affected. Same reasons as Admissable one. Maybe it can't find the solution.