1. Implement an AStarSearch Class. Implement the search function by modifying the Uniform Cost Search algorithm. Note that the heuristic function h is to be defined in the EightPuzzle.py file.

2. In EightPuzzle.py, implement the heuristic function that counts the number of misplaced tiles. def h(self, state):

• Document the number of nodes that are searched and total time using a configuration of your choosing.

• Describe why the AStar algorithm is much more effective in finding an optimal solution compared to Uniform Cost Search.

For 24 path length initial start (2, 3, 1, 8, 0, 6, 5, 7, 4)

Nodes searched with UCS search: 130983

Time Spent with UCS search: 3.0534839630126953

Nodes searched with A Star search: 52028

Time Spent with A Star search: 1.1883792877197266

UCS search was fairly lost in this puzzle as all path costs were the same, thus it could do little better than breath-first search. A Star Search on the other hand is directed by its heuristic and so it does not need to check nearly as many nodes as it recognized beforehand that some of them will not be helpful.

3. Comment out your implementation for the heuristic for misplaced tiles, and implement a heuristic that counts the total Manhattan distance.

• Document the number of nodes that are searched and total time using a configuration of your choosing This should be the same configuration used in part 2 and should show an improvement.

• Describe why this heuristic results in an improved performance.

For 24 path length initial start (2, 3, 1, 8, 0, 6, 5, 7, 4)

Nodes searched with UCS search: 130983

Time Spent with UCS search: 2.884113311767578

Nodes searched with A Star search Misplaced Tiles Heuristic: 52028

Time Spent with A Star search: 1.1883792877197266

Nodes searched with A Star search Manhattan Distance Heuristic: 13813

Time Spent with A Star search: 0.4343845844268799

The Manhattan Distance Heuristic provides a much better approximation for how many moves we will actually have to make and it is possible to get a state that has more tiles in the right place but overall dose not decrease the number of moves to make it to the goal state. This allows the Manhattan Distance Heuristic to be more directed in its search for the goal state, it can avoid even more unnecessary searth paths then the last herusic allowing it to find the solution in a fraction of the time.