Homework 1

Python Program

Code Files: Gamestate.py, Search.py
Input Files: input.txt

The Gamestate.py implements the class for storing the state of the game. The file Search.py is the program that is to be run. It imports the class and implements the 3 search algorithms: Breadth first search, Depth first search and A*

The initial state of the game is given in the file input.txt. The command to execute the program is

```
python Search.py --input input.txt
```

When the program is run, it displays the results of each of the search algorithm along with the details for the same.

Question 4

The stats for the program, for the initial state given as below:

1,2,5
3,4,0
6,7,8

Parameters	BFS	DFS	A*
Cost of path	3	3	3
Nodes Expanded	12	97	12
Maximum depth	5	63	5
Memory	280 Bytes	3528 Bytes	280 Bytes
Running Time	7 milliseconds	63 milliseconds	6 milliseconds

The stats for the program, for the initial state given as below:

1,2,5
0,3,4
6,7,8

Parameters	**BFS**	**DFS**	**A***
Cost of path	5	5	5
Nodes Expanded	196	545	87
Maximum depth	69	351	32
Memory	3864 Bytes	19656 Bytes	1792 Bytes
Running Time	132 milliseconds	400 milliseconds	63 milliseconds

The results shown In the above tables summarize that the A* based on heuristic is the best performer in all aspects. It also shows that BFS performs better than that of DFS in memory and execution time. This is because the solutions are shallow, the branching factor is less and hence BFS ends up using less memory.

Question 5

The heuristic chosen here is “Misplaced Tiles”. This is basically keeping count of the number of misplaced tiles as compared to that of the Goal state. The heuristic is implemented by attaching a priority to the nodes, so that instead of random or in-order selection of next node to be explored, the AI now selects based on priority. The heuristic calculates the number of tiles that are misplaced (how close is the current state to that of the goal state) and assigns this as priority to the nodes that need to be explored.

This efficiently prunes the search tree, reducing the number of nodes to be explored; which reduces memory, and also makes better choices of next node to be explored; which leads us to the solution faster.

Since, the heuristic function simply calculates the closeness of the state to the goal state by the number of tiles misplaced it never over estimates the cost to the goal. Since this heuristic function never over estimates the cost of reaching the goal, it is an admissible heuristic function.

Question 6

The knowledge representation for this AI involves representing the game state similar to that of actual play. The game states are the position of the numbers in each index. The empty tile is represented by 0. The agent choice is to decide where to move the empty tile next. This choice is done by checking the neighbors of the empty tile for legal moves of UP/DOWN/LEFT/RIGHT. If it is a legal move, then the move is performed and its implications are calculated and followed upon, otherwise an alternative move is chosen.

This representation is as close to the actual game as possible and the game play is also similar to that of actual play and rules. It is compact, natural and maintainable. It also holds all information required to solve the problem.