Comp 3710 Artificial Intelligence Concepts.

Assignment 2 (Points 10)

Due on 27/02/2021 Before 11:59pm

Part I (Points 4)

Recall assignment 1-part I, you implemented various uninformed searching algorithms and returned traversal paths and exact paths for a given graph. Here, you will consider the same scenario and implement A* searching algorithm, which is an informed search algorithm and return a traversal path and exact path of a given graph. Then, compare the cost from A* search algorithm with other uninformed search algorithms on the graphs (a) and (b) (refer assignment 1).

Heuristic functions:

States	h(n)
S	6
A	4
B	3
C	3
D	1
G	0

Graph a

States	h(n)
S	14
A	7
B	10
C	4
D	2
E	4
G	0

Graph b

Part II (Points 6)

The 8-puzzle is a toy problem that we discussed briefly in class. Your task is to write the following programs to solve a randomly generated 8-puzzle.

1. Write following helper functions: (**1.5 points**)
    a. generate_random_8puzzle () – to return a random 8-puzzle problem that is solvable.
    b. check_solvable(state) – to return whether the generated 8-puzzle is solvable or not.
    Hint: For 8-puzzle problem, if an inversion is any pair of tiles i and j where i < j but i appears after j in row-major order. Consider following two different 8-puzzle problem instances.

```
1 2 3
4 5 6
8 7
```

Order of tiles: 1 2 3 4 5 6 8 7
inversions = 1
(8-7)
unsolvable

```
  1 3
4 2 5
7 8 6
```

Order of tiles: 1 3 4 2 5 7 8 6
inversions = 4
(3-2, 4-2, 7-6, 8-6)
solvable

if an 8-puzzle instance has an odd number of inversions, then it is not solvable. If it has even number of inversions, it is solvable.

c. display_8puzzle(state) -to print a state of a puzzle in the following format.

```
*  1  3
4  2  5
7  8  6
```

2. Write A* search algorithm to solve 8-puzzle problems by using the following heuristic functions. (**3 points**)
   a. the misplaced tile heuristic
   b. the Manhattan distance heuristic
   c. the max of the misplaced tile heuristic and the Manhattan distance heuristic

3. Generate **five** random solvable 8-puzzle instances and solve them using all three methods implemented in the previous question. For each instance, (**1.5 points**)
   a. Record the total run time (in seconds) of each method
   b. Cost to reach the solution (i.e., number of tiles moved)

   Summarize your result as a table and state the best performed algorithm with a suitable reason.

NOTE: The "goal" state of the 8-puzzle should be in order – 1 2 3 4 5 6 7 8 9. However, the blank tile can be placed anywhere on the board. The following are few acceptable goal states.

```
   1  2
3  4  5
6  7  8
```

```
1  2  3
4  5  6
7  8
```

```
1  2  3
8     4
7  6  5
```

Submission should include the following:

- The source codes
- A PDF document with the following information:
  - Screenshots of your test result or Makefile - include required information of each screenshot
  - The comparison results as a table.

*****