Assignment – 2 (CLO-3)

Problem description:

For this assignment, you will Implement A* algorithm for the following problem:

- 8 Puzzle

The pseudocode for A* algorithm is provided at the end of this assignment document. Your task is to use the algorithm and implement it in Python. You can use any of the following heuristic functions:

1. h1 = the number of misplaced tiles
2. h2 = the sum of the distances of the tiles from their goal positions

Input:

- Your program does not need any input from the user. Your program should specify a randomly generated configuration of the puzzle in the following format:

  1 2 4
  0 5 6
  8 3 7

  The goal state is:

  0 1 2
  3 4 5
  6 7 8

  Where 0 represents the empty tile

Output:

The output should show the sequence of STEPS (moves) and the number of STEPS (moves) your algorithm took to reach the GOAL state from the START state.

Deliverable:

- You should upload a zip file containing the assignment folder. The folder should contain the following:
  1. Your python code file.
  2. You can also use Jupyter Notebook.
  3. Screenshots of your program output.
  4. Properly commented code.

Note:

- This is an individual assignment (no groups).
- Deadline: **11:59 PM, 4th May, 2021**. No late submissions allowed/accepted.
- Make sure all files have your Names and Enrolment numbers.

A* Algorithm Pseudocode:

```
make an empty closed list
   while (the destination node has not been reached):
       consider the node with the lowest f score in the open list
       if (this node is our destination node) :
           we are finished
       if not:
           put the current node in the closed list and look at all of its neighbors
           for (each neighbor of the current node):
               if (neighbor has lower g value than current and is in the closed list) :
                   replace the neighbor with the new, lower, g value
                   current node is now the neighbor's parent
               else if (current g value is lower and this neighbor is in the open list ) :
                   replace the neighbor with the new, lower, g value
                   change the neighbor's parent to our current node

               else if this neighbor is not in both lists:
                   add it to the open list and set its g
```