CS 4613 **Spring 2022**
Project 1: 8-Puzzle Problem **E. K. Wong**

Total # points = 100.

Project Description: Implement the *A** search algorithm with *graph search* (with no repeated states) for solving the 8-puzzle problem with heuristic functions $h_1(n)$ and $h_2(n)$ as described below.

8-puzzle problem: On a 3 x 3 board there are 8 tiles numbered from 1 to 8 and a blank position. A tile can slide into the blank position if it is horizontally or vertically adjacent to the blank position. Given a start board configuration (initial state) and a goal board configuration (goal state), find a move sequence with a minimum number of moves to reach the goal configuration from the start configuration. An example pair of initial state and goal state is given in the figure below.

8	1	3
	2	4
7	6	5

Initial state

1	2	3
8		4
7	6	5

Goal state

Heuristic function $h_1(n)$ is *the sum of Manhattan distances of the tiles from their goal positions.* Heuristic function $h_2(n)$ is the *Nilsson's sequence score* defined as

$$h_2(n) = P(n) + 3\,S(n)$$

where $P(n)$ is the sum of Manhattan distances of the tiles from their goal positions and $S(n)$ is the sum of the scores obtained by (1) going around the tiles along the border in a clockwise direction, allotting a score of 2 for every tile not followed by its proper successor and 0 otherwise, and (2) allotting a score of 1 to the center tile if it does not match the center tile of the goal state and 0 otherwise. As an example, in the initial state above, $P(n) = 3$. $S(n)$ is obtained by going around the tiles along the border in a clockwise direction. Starting with the upper left corner and go around, we see that tile 1 in the initial state is followed by 3 but it should be followed by 2 as in the goal state. We therefore allot a score of 2 to tile 1; similarly we allot a score of 2 to tile 7 since 7 in the initial state is followed by the blank position whereas it is followed by 8 in the goal state. The center tile 2 in the initial state does not match the center tile of the goal state (a blank) and is allotted a score of 1. Therefore, $S(n) = 2 + 2 + 1 = 5$ and $h_2(n) = 3 + 3 \times 5 = 18$. Heuristic function $h_2(n)$ is an *inadmissible* function that does not guarantee optimal solutions in the A* search but it often generates fewer nodes in the search process.

Your program will read in the initial and goal states from an input file and then generates an output file that contains the solution.

You can work on the project by yourself or form a team of two students to work on the project. You can discuss with your classmates on how to do the project but every team is expected to write their own code and submit their own project.

Input and output file formats: Your program will read in the initial and goal states from a text file that contains 7 lines as shown in Figure 1 below. Lines 1 to 3 contain the tile pattern for the initial state and lines 5 to 7 contain the tile pattern for the goal state. Line 4 is a blank line. Variables *n* and *m* are integers that range from 0 to 8, with 0 representing the blank position and integers 1 to 8 representing tile numbers.

Your program will produce an output file that contains 12 lines as shown in Figure 2 below. Lines 1 to 3 and lines 5 to 7 contain the tile patterns for the initial and goal states as given in the input file. Lines 4 and 8 are blank lines. Line 9 is the depth level *d* of the shallowest goal node as found by your search algorithm (assume the root node is at level 0.) Line 10 is the total number of nodes *N* generated by the A* algorithm (including the root node.) Line 11 contains the solution that you have found. The solution is a sequence of actions (from root node to goal node) represented by the A's in line 11, separated by spaces. Each A is a character from the set {L, R, U, D}, representing the left, right, up and down movements of the blank position. Line 12 contains the $f(n)$ values of the nodes along the solution path from the root node to the goal node, separated by spaces. There should be d number of A values in line 11 and $d+1$ number of f values in line 12.

Testing your program: Three input files will be provided on *Brightspace* for you to test your program (All three input files are solvable; i.e., they have solutions.) For each input file, produce an output file for each of the two heuristic functions $h_1(n)$ and $h_2(n)$, for a total of six output files.

Recommended languages: Python, C++/C and Java. If you would like to use a language other than these three, send me an email first.

Submit on *Brightspace* by the due date:

1. Your source code file. Put comments in your source code to make it easier for someone else to read your program. Points will be taken off if you do not have comments in your source code.
2. The six output files generated by your program for test input files 1 to 3. Name the output files generated by using heuristic $h_1(n)$ as *output1h1.txt*, *output2h1.t*xt and *output3h1.txt*; and name the output files generated by using heuristic $h_2(n)$ as *output1h2.txt*, *output2h2.txt* and *output3h2.txt*.
3. A PDF file that contains instructions on how to run your program. If your program requires compilation, instructions on how to compile your program should also be provided. Also, copy and paste your output files and your source code onto the PDF file (to make it easier for us to grade your project.) This is in addition to the source code file and output files that you have to hand in separately, as described in (1) and (2) above. For each output file, clearly indicate which heuristic function (h_1 or h_2) was used to generate the output.

If you work in a team of two, only one partner needs to submit the project on Brightspace but please write down both partners' names on the source code and the PDF report. (**Please do not submit two projects if you work in a team of two.**)

n n n
n n n
n n n

m m m
m m m
m m m

Figure 1. Input file format (7 lines.)

n n n
n n n
n n n

m m m
m m m
m m m

d
N
A A A A A A A ……
f f f f f f f f ……..

Figure 2. Output file format (12 lines.)