CS 4613 **Spring 2019**
Project 1: 8 Puzzle Problem **E. K. Wong**

Total # points = 100. Figures 1 and 2 are on page 2.

Project Description: Implement the GRAPH-SEARCH Algorithm with *A* Search Strategy* for solving the 8-puzzle problem. Write your program so that the user can choose one of the following heuristic functions to use in the evaluation function: (1) *sum of Manhattan distances of tiles from their goal position* and (2) *sum of Manhattan distances + 2× # linear conflicts*. You are to work on the project by yourself. You are allowed to discuss with your classmates how to do the project but everyone is expected to write her/his own program.

Input and output file formats: Your program will read in the initial and goal states from an ASCII file that contains seven 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. *n* and *m* are integers that range from 0 to 8. Integer 0 represents the blank position and integers 1 to 8 represent tile numbers. Your program will produce an output ASCII file that contains 11 lines as shown in Figure 2. 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 goal node in your search tree (assume the root node is at level 0.) Line 10 is the total number of nodes *N* generated in your search tree (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. Each A in line 11 is a character from the set {L, R, U, D}, where L, R, U and D represent 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 root node to the goal node. There should be *d* number of A values in line 12 and *d*+1 number of *f* values in line 13.

Testing your program: I will provide a few test input files on NYU Classes for you to test your program but you can also create your own test files. For each test file, run your program with heuristic (1) above to produce an output file and then run with heuristic (2) to produce a second output file. A 15 min time slot will also be set up during class time for you to run your program on test files that will be provided to you in class at that time (Details to be announced later.)

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

Hand in by due date on NYU Classes:

1. A text file that contains the source code. 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. Output ASCII files for the test files that will be provided on NYU Classes. For each test file, produce one output file using heuristic (1) and another using heuristic (2) above.
3. An MS Words or PDF file that contains instructions on how to run your program. Also, copy and paste the output ASCII files and your source code onto the Words or PDF file.

```
n n n
n n n
n n n

m m m
m m m
m m m
```

Figure 1. Input file format.

```
**************************
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 f ……..
```

Figure 2. Output file format.