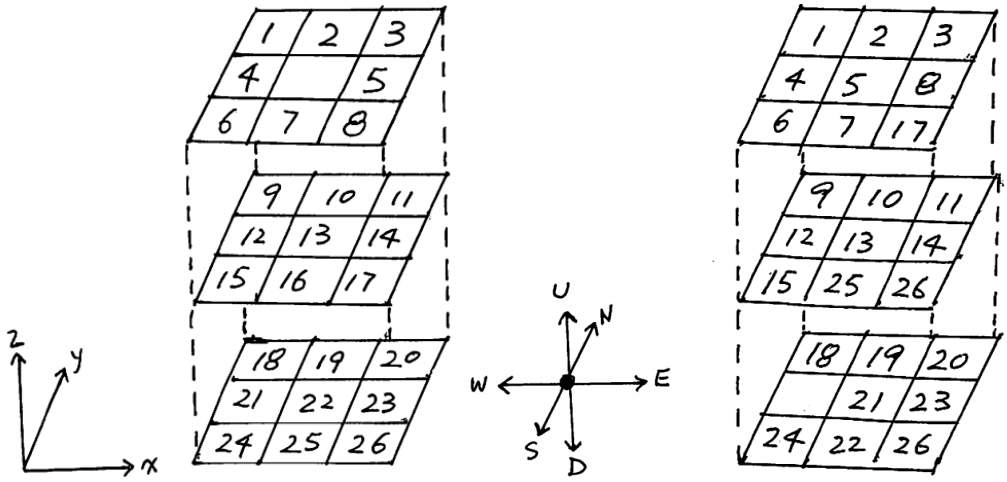
CS 4613 **Fall 2023**
Project 1: 26-Puzzle Problem **E. K. Wong**

Total # points = 100.

Project Description: Implement the A* search algorithm with graph search (no repeated states) for solving the 26-puzzle problem as described below. Use h(n)= Sum of Manhattan distances of the tiles from their goal positions as heuristic function. Your program will read in the initial and goal states from an input file and then generate an output file that contains the solution.

The 26-puzzle problem: The game board consists of three 3 x 3 grids stacked together as shown in Figure 1 below. There are 26 tiles, numbered 1 to 26, and a blank position. A tile can move into the blank position if it is adjacent to the blank position in the x, y or z directions, as shown in the figure. We can define six virtual moves (actions) for the blank position: *East* (E), *West* (W), *North* (N), *South* (S), *Up* (U) and *Down* (D) (see figure below.) Given an initial state, the goal is to find a move sequence with a minimum number of moves to reach a given goal state. For example, the optimal solution for the initial and goal states in Figure 1 is ESDWDNW.

Note: Since the game board is 3D, the Manhattan distance is defined as $Dist = |\Delta x| + |\Delta y| + |\Delta z|$.



(a) Initial state (b) Actions (c) Goal state

Figure 1. The 26-puzzle problem

Teammate: 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 files: Your program will read in the values for the initial state and goal state from a text file that contains 23 lines as shown in Figure 2 below. Lines 1 to 11 contain the tile patterns for the three layers of the initial state, with the layers separated by blank lines. Line 12 is a blank line. Lines 13 to 23 contain the three layers of the goal state, with the layers separated by blank lines. *n* and *m* are integers that range from 0 to 26, representing the blank position (0) and the tile numbers (1-26.)

Your program will produce an output file that contains 28 lines as shown in Figure 2 below. Lines 1 to 23 contain the tile patterns for the initial state and goal state. You can simply copy lines 1 to 23 from the input file to the output file. Line 24 is a blank line. Line 25 is the depth level *d* of the shallowest goal node as found by the A* algorithm (assume the root node is at level 0.) Line 26 is

the total number of nodes *N* generated in your tree (including the root node.) Line 27 contains the solution (a sequence of actions from root node to goal node) represented by A's. The A's are separated by blank spaces. Each A is a character from the set {E, W, N, S, U, D}, representing the *East, West, North, South, Up* and *Down* movements of the blank position. Line 28 contains the $f(n)$ values of the nodes along the solution path, from the root node to the goal node, separated by blank spaces. There should be d number of A values in line 27 and $d+1$ number of f values in line 28.

Testing your program: Three input files will be provided on Brightspace for you to test your program. All input files are solvable; i.e., they have solutions.

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

Submit on Brightspace:

- 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.
- The output files generated by your program for the input test files.
- 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 outputs 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 items (1) and (2) above.

If you work in a team of two, only one partner needs to submit but please write both partners' names on the source code and the PDF report.

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

n n n
n n n
n n n

n n n
n n n
n n n

m m m
m m m
m m m

m m m
m m m
m m m

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

Figure 2. Input file format (23 lines.)

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

n n n
n n n
n n n

n n n
n n n
n n n

m m m
m m m
m m m

m m m
m m m
m m m

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
```

Figure 3. Output file format (28 lines.)