

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 15-puzzle problem as described below. Use *sum of Manhattan distances of tiles from their goal positions* as heuristic function. [Reminder: the GRAPH-SEARCH Algorithm does not allow repeated states.]

15-puzzle problem: On a 4 x 4 board there are 15 tiles numbered from 1 to 15 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 and a goal board configuration, find a sequence of moves to reach the goal configuration from the start configuration. The goal is to find a solution with minimum **number of moves**.

You are to work on this project by yourself. You can discuss with your classmates on how to do the project but everyone is expected to write her/his own code and submit their own program.

Input and output file format: Your program will read in the initial and goal states from a text file that contains nine lines as shown in Figure 1 below. Lines 1 to 4 contain the tile pattern for the initial state and lines 6 to 9 contain the tile pattern for the goal state. Line 5 is a blank line. n and m are integers that range from 0 to 15. Integer 0 represents a blank position and integers 1 to 15 represent tile numbers. Your program will produce an output text file that contains 14 lines as shown in Figure 2 below. Lines 1 to 4 and lines 6 to 9 contain the tile patterns for the initial and goal states as given in the input file. Lines 5 and 10 are blank lines. Line 11 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 12 is the total number of nodes N generated in your tree (including the root node.) Line 13 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 13. Each A is a character from the set {L, R, U, D}, which represents the left, right, up and down movements of the blank position. Line 14 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 13 and $d+1$ number of f values in line 14.

Testing your program: Four input test files will be provided on NYU Classes for you to test your program.

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 NYU Classes by due date:

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. The **output text files** generated by your program for test input files 2 to 4.
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 the output text 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 text files that you have to hand in as described in (1) and (2) above.

```
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
```

Figure 1. Input file format.

```
*****
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

d
N
A A A A A A .....
f f f f f f f f .....
```

Figure 2. Output file format.