Artificial Intelligence **1st Semester, 2021-2022**

Machine Problem

Write a program in C/C++ implementing a blind search strategy, i.e. Iterative Deepening Search(IDS) and a heuristic search strategy, i.e. A* Search with graph search to solve the 8-puzzle problem using Manhattan distance as the heuristic. Your program should use the board configuration below as the goal state and let the user input the initial/start board configuration.

1	**2**	**3**
8		**4**
7	**6**	**5**

It should output the following for both the IDS and A* Search:

- solution path(corresponds to the moves needed to reach the goal): e.g. [Up-Left-Left-Right]
- solution cost(# of moves in the solution): 4
- number of nodes expanded before reaching the goal
- running time

Submit the following:

1. Source code of your interactive program. Make sure that it can be compiled on gcc, so please use only standard functions.
    - Make sure that your name/s will also appear on your source codes as comments.
    - Although there maybe available codes on the internet for the 8-puzzle, make sure that for the A* Search, your codes follow the pseudocode given in the lecture. As much as possible, write your code from scratch given the pseudocode described.

2. Documentation that includes the following:
    - a detailed discussion/explanation of your implementation or source code and the participation/contribution of each member in making the machine project.
    - the analysis and comparison of the performance of the two algorithms. Use the table provided in the next page to show your analysis and comparison.

There will be a short demonstration/presentation to defend your program which can be conducted through Zoom or Google Meet.

You can work on this project by group with 3-4 members. Work independently from your other classmates to avoid having the same output. Outputs that are almost identical will not be accepted.

<table>
<tr><th>Initial State</th><th></th><th>IDS</th><th>A*</th></tr>
<tr><td rowspan="4"><b>Easy</b><table><tr><td>1</td><td>3</td><td>4</td></tr><tr><td>8</td><td>6</td><td>2</td></tr><tr><td>7</td><td></td><td>5</td></tr></table></td><td>Solution path</td><td></td><td></td></tr>
<tr><td>Solution cost</td><td></td><td></td></tr>
<tr><td>Number of nodes expanded</td><td></td><td></td></tr>
<tr><td>Running Time</td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td></tr>
<tr><td rowspan="4"><b>Medium</b><table><tr><td>2</td><td>8</td><td>1</td></tr><tr><td></td><td>4</td><td>3</td></tr><tr><td>7</td><td>6</td><td>5</td></tr></table></td><td>Solution path</td><td></td><td></td></tr>
<tr><td>Solution cost</td><td></td><td></td></tr>
<tr><td>Number of nodes expanded</td><td></td><td></td></tr>
<tr><td>Running Time</td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td></tr>
<tr><td rowspan="4"><b>Hard</b><table><tr><td>2</td><td>8</td><td>1</td></tr><tr><td>4</td><td>6</td><td>3</td></tr><tr><td>7</td><td>5</td><td></td></tr></table></td><td>Solution path</td><td></td><td></td></tr>
<tr><td>Solution cost</td><td></td><td></td></tr>
<tr><td>Number of nodes expanded</td><td></td><td></td></tr>
<tr><td>Running Time</td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td></tr>
<tr><td rowspan="4"><b>Worst</b><table><tr><td>5</td><td>6</td><td>7</td></tr><tr><td>4</td><td></td><td>8</td></tr><tr><td>3</td><td>2</td><td>1</td></tr></table></td><td>Solution path</td><td></td><td></td></tr>
<tr><td>Solution cost</td><td></td><td></td></tr>
<tr><td>Number of nodes expanded</td><td></td><td></td></tr>
<tr><td>Running Time</td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td></tr>
<tr><td rowspan="4"><b>Your preferred initial configuration</b></td><td>Solution path</td><td></td><td></td></tr>
<tr><td>Solution cost</td><td></td><td></td></tr>
<tr><td>Number of nodes expanded</td><td></td><td></td></tr>
<tr><td>Running Time</td><td></td><td></td></tr>
</table>