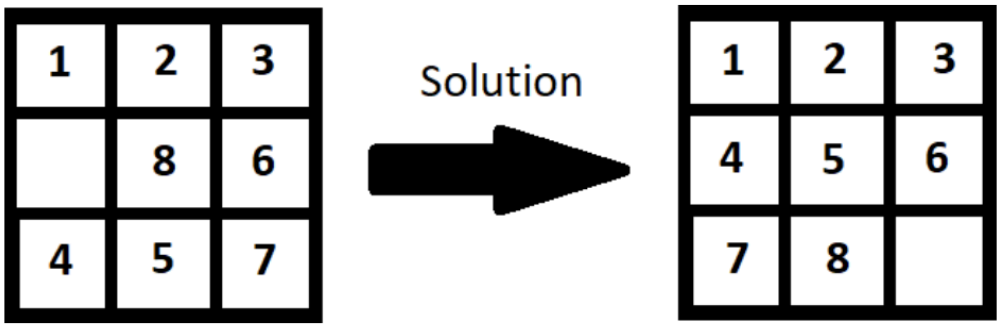
Fall 2024 COSC 3P71 Artificial Intelligence: Assignment 1

Instructor: Reginald McLean (rmclean@brocku.ca)
Course Coordinator: Zachary McGovarin (zm19hc@brocku.ca)
Tutorial Leader: Zachary McGovarin
Teaching Assistants: Rasa Khosrowshahli (aq23mb@brocku.ca), Tyler McDonald (tm21cy@brocku.ca), Michael Eshun (me22eh@brocku.ca)

Assigned date: September 30th, 2024
Due Date: October 18th, 2024 at 11:59 PM
Goal: Implement a program to solve the **Sliding Number** puzzles using various search algorithms
Programming Languages: Any language available in the labs. Recommendation: C#, Python, Java
Sliding Numbers Puzzle:
A sliding numbers puzzle is a 2D combination puzzle, usually containing either 8 or 15 numbers and one blank space. The puzzle is to rearrange the numbers in the puzzle so that they are all in numerical order with the blank space at the end. This is done by sliding one number at a time into the blank space to move the tiles around.



Your task for this assignment will be to write a program that can solve such sliding numbers puzzles. The puzzle will be read in via a text file and can be either a 3x3 or 4x4 puzzle. Note that not all possible puzzles are solvable, but the examples provided all do have a solution. A text-based interface is sufficient for this assignment, though if you choose to implement a graphical interface that will also be accepted. Your program will output the following information each run: Initial puzzle, solved puzzle, and the moves needed to go from beginning to end (it is up to you to decide how you wish to output this information).

To solve the puzzles, you will implement several different search strategies. While the problem seems simple on the surface, a brute force search has a computational complexity of O((n + 1)!), where n is the number of tiles in the puzzle (i.e., the above puzzle has 9! possible positions). The search algorithms you will implement are as follows:

Iterative Deepening Depth First Search: This strategy uses a Depth First Search with an incremental max depth and is also guaranteed to find a solution in the least possible number of moves.
A* Heuristic Search: This strategy implements the A* search and will make use of heuristic evaluations to improve efficiency. A* is guaranteed to find an optimal solution provided that the implemented heuristic is admissible.
Along with the implementation, you will submit a short write up. This write up will include:

1. A* with the two heuristics covered in class
2. A* with a heuristic of your design/research (it can be an original idea, or a heuristic that already exists that you find online)
3. A description of your heuristic implemented for your A* search.
4. A brief section for each implemented search strategy about how it performed compared to the other strategies.
5. A brief section for each implemented heuristic function and how they compare to each other
6. An explanation of how to run and use the program, how to interpret the output (if needed), and any additional information you feel is needed to inform the marker of.

Note: Bonus marks will be awarded to those that implement the Iterative Deepening A* (assignment is worth 100% without the bonus).

Hints:

- When implementing A* search, make sure to spend a good amount of time on the heuristic function. You are welcome to research already existing heuristics for this problem and are encouraged to also experiment with your own ideas.
- It is normal for some puzzles to take a considerable amount of time to solve, if your program takes a while to run it does not mean that you have done something wrong.

Hand in and submission notes:

- Submission will be done electronically through Brightspace. Submission will include all your commented source code and your write up.
- Unity is not allowed for this assignment by any students. We recommend using at least Java 8, Python 3.6, or C# . Your source code should be presented in a format that can be easily compiled. All assignments must include detailed instructions for the marker to compile and run the submission. Failure to provide adequate explanation and documentation may result in a submission not being graded.
- Please note that the virtual COMMONS is available to all students at Brock. You are not required, but if you prefer to (or need to) you can use the virtual COMMONS instead of a personal computer. Machines in the virtual COMMONS have IDEs for Java, Python, and C#.
- Any questions/concerns etc., regarding the grading of any assignment MUST be raised within 7 days of graded assignment hand-back. In this case, please send your concerns/questions to Course Coordinator Zachary McGovarin

Example Puzzles on next page

Example Puzzles

```
3
1   2   3
X   8   6
4   5   7

3
X   2   5
7   3   8
4   6   1

4
1   2   8   3
5   10  6   4
9   14  7   12
13  X   11  15

4
5   8   6   3
2   1   11  X
13  10  14  4
7   9   15  12
```