Project: 1 CS 170. Introduction to Artificial Intelligence

Due Date: Nov 6th: 2019

The Eight Puzzle

For this project I want you to write a program that solves the eight-puzzle. You will solve it using

1) Uniform Cost Search[1]
2) A* with the Misplaced Tile heuristic.
3) A* with the Manhattan Distance heuristic.

You may use any language, but the variable and function names for main "driver" program should approximately match the peusdocode in the book. If you have a good reason to ignore this directive, let me know in advance.

```
function general-search(problem, QUEUEING-FUNCTION)
nodes = MAKE-QUEUE(MAKE-NODE(problem.INITIAL-STATE))
loop do
 if EMPTY(nodes) then return "failure"
  node = REMOVE-FRONT(nodes)
 if problem.GOAL-TEST(node.STATE) succeeds then return node
   nodes = QUEUEING-FUNCTION(nodes, EXPAND(node, problem.OPERATORS))
 end
```

The general search algorithm.

I expect the code to be meaningfully commented, nicely indented etc. This is not a software engineering class, but if I cannot clearly review and understand your code. I will strongly lean towards a lower grade.

The code should be kept general as possible. In other words, your code should require only a modicum of effort to change to solve the 15-puzzle, or the 25-puzzle etc.

You can hardcode an initial state for testing purposes. But I want to be able to enter an arbitrary initial state. So sometime along the lines of the interface on the next page would be nice.

You may use some predefined utility routines, for example sorting routines or queue manipulation functions. However, I expect all the major code to be original. You must document any book, webpage, person or other resources you consult in doing this project (see the first day's handout).

You may consult colleagues at a high level, discussing ways to implement the tree data structure for example. But you may **not** share code. At most, you might illustrate something to a colleague with pseudocode.

You will hand in

1) A coversheet.
2) A printout of your code.
   (just the first and last page, don't kill trees)
   A trace of the Manhattan distance A* on the following problem. (just the first and last page, don't kill trees)

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

3) A two to five-page report which summaries your findings. You might want to compare the algorithms on several random puzzles, summarize the results with graphs and make comments on the utility of the different heuristics etc. The only meaningful comparisons for the algorithms are time (number of nodes expanded) and space (the maximum size of the queue).

[1] Note that Uniform Cost Search is just A* with $h(n)$ hardcoded to equal zero.

You **must** keep the evolving versions of your code, so that, if necessary you can demonstrate to the course staff how you went about solving this problem (in other words, we may ask you to prove that you did the work, rather than copy it from somewhere).
In addition, I will randomly select ten students to demonstrate their program to me, in my office.

You can use a simple text line interface or a more sophisticated GUI (but don't waste time making it pretty unless you are sure it works and you have lots free time). However your program should generate a trace like the one below, so that it can be tested.

```
Welcome to Bertie Woosters 8-puzzle solver.
Type "1" to use a default puzzle, or "2" to enter your own puzzle.
2
        Enter your puzzle, use a zero to represent the blank
        Enter the first row, use space or tabs between numbers      1 2 3
        Enter the second row, use space or tabs between numbers     4 8 0
        Enter the third row, use space or tabs between numbers      7 6 5

        Enter your choice of algorithm
            1. Uniform Cost Search
            2. A* with the Misplaced Tile heuristic.
            3. A* with the Manhattan distance heuristic.

            3

Expanding state 1 2 3
                4 8 b
                7 6 5

The best state to expand with a g(n) = 1 and h(n) = 4 is…
                1 2 3
                4 8 5
                7 6 b    Expanding this node…

The best state to expand with a g(n) = 2 and h(n) = 3 is…
                1 2 3
                4 8 5
                7 b 6    Expanding this node…

{steps omitted, you can omit some steps in your print out, to save paper, Eamonn}


The best state to expand with a g(n) = 4 and h(n) = 1 is…
                1 2 3
                4 5 6
                7 b 8    Expanding this node…


Goal!!

To solve this problem the search algorithm expanded a total of 123 nodes.
The maximum number of nodes in the queue at any one time was 45.
The depth of the goal node was 5
```

These numbers are made up, your results may differ.