CSCE A405 Programming Assignment 1

**Due Thursday, September 20, 2018, at 11:59 PM**

Purpose: The goal of this assignment is to gain an understanding of the advantages of using informed search techniques.

Requirements: In this assignment, you are required to write a program that solves a variety of 8-Puzzle problems using each of the following search techniques:

1. Breadth-First Search
2. Greedy Best-First Search using the Manhattan Distance heuristic
3. A\* Search using the Misplaced Tiles heuristic
4. A\* Search using the Manhattan Distance heuristic

Note that these search techniques differ only in how they evaluate nodes: A\* uses f(n) = g(n) + h(n), while Breadth-First Search uses f(n) = g(n) and Greedy Best-First Search uses f(n) = h(n). Your program should allow the user to interactively input the START and GOAL states. Your program should then attempt to identify a path from the START state to the GOAL state, using each of the search techniques listed above. During each search, your program should keep track of the total number of nodes expanded; the maximum number of nodes represented in the search space at any particular time during the search; and the length of the solution path. Once a solution has been identified, your program should output the each of these values, and display the solution path as a sequence of states from START to GOAL.

*Work in teams of one to three students.* Programs should be written in a high-level language such as Java, Visual Basic, or C/C++. Design your solution such that the details of each search technique are hidden within the implementation of the corresponding search function, and are thus effectively isolated from the rest of your program. All four search techniques should be integrated into a single program. Due to the magnitude of this assignment, I strongly recommend that you begin designing, implementing, testing, debugging, and validating the correct operation of your program as soon as possible!

The solution space of the 8-Puzzle is symmetrically partitioned according to even or odd PARITY. The parity of a given state can be calculated by counting the number of pairs of tiles in the “wrong” order, assuming the all of the tiles in the following configuration are in the “right” order:

1 2 3

4 5 6

7 8

A solution path exists from a START state to a GOAL state if and only if either both states have odd parity or both states have even parity.

Debug your program using (START, GOAL) state pairs for which a relatively simple solution exists. Once you are satisfied that each search technique works correctly, test your program for a large number of problems having a wide variety of minimum solution path lengths, and note the relationship between the number of nodes expanded during each search and the length of the solution path. Your test set should also include several problems for which the minimum solution path lengths are equal; note the differences in the performance of each search technique, and identify characteristics that make one problem more difficult to solve than another. Record test results for each search technique.

*Create a jointly written report summarizing your results.* Clearly identify the name and preferred email address of each team member on the cover page of your report. Your report should describe each search technique; provide a summary of test results for each search technique for several representative runs; and identify properties of 8-Puzzle problems that make them harder or easier to solve using each technique. Conclude your report by comparing the overall performance of each of the four techniques.

Submit your report, source code, and executable (if applicable) via Blackboard. I will test your program interactively, read your report, and send your grade with appropriate comments via electronic mail.