CSC361: Artificial Intelligence **Due: Dec. 2nd, 2018**

Computer Science Department 2018, Fall

King Saud University Project: Search

This is the first milestone of your project, the project carries 10% *of your grade - You will work in groups of 3 students*

You will develop an agent that solves the 8-puzzle problem, it will receive the initial state of the puzzle and figure out a sequence of actions that leads to the solved puzzle

a) **Input:**

- The input of your program is a file with a sequence of 9 numbers, where 0 represents the empty space
- The search algorithm to be used (BFS, DFS, Greedy, A*)

Sequence to Puzzle: As mentioned before, the file contains a sequence of numbers, these should correspond to tiles, in row by row order from left to right.
Example of an input file:

```
1, 0, 2, 3, 4, 5, 6, 7, 8
```

corresponds to the following puzzle:

1		2
3	4	5
6	7	8

b) **Output:** Your program will run the specified algorithm and output:

- The solution found by the search strategy- as sequence of steps from initial state e.g. 'up, up, left.. etc', where the goal state is:

1	2	3
4	5	6
7	8	

- As an option, it should display the sequence of states in the path to goal
- The cost of the solution
- The time it consumed (in ms)
- The number of nodes generated
- The number of nodes expanded

c) **How to solve the problem** Start by formulating the problem as a search problem. This will help you figure out what functions you need to implement.

You are asked to solve this problem in different search strategies, but remember, you do not have to write a different search algorithm for each one. The only thing that changes is the data structure for the fringe and evaluation function for Informed Search.

Hint: Avoid repetition in your search, to achieve that do not forget to include a closed set. It might be easiest for you to use a two-dimensional array to represent the puzzle.

Propose two simple admissible heuristic functions, and compare between them. Bonus points will be granted for novel heuristic functions.

You do not need to use any external libraries, except for measuring the time. You have the option to implement this using either Java or Python

While it might be tempting to divide the work amongst group members, where each member works individually, team-mates are encouraged to work together. This will make the project more entertaining and help you learn from each other!

d) **Deliverables**

- The source code (the project folder). Make sure your code is well organized and documented.
- A report that includes the following sections:
  - Cover page
  - An overview of the implementation
  - The main functions (or classes) used in your program and how they interact with each other
  - A comparison of the performance of the search algorithms
  - The admissible heuristics found, with commentary on their quality and how they affect the search
  - For each search algorithm implemented provide an example of the output of your program
  - Describe challenges faced and how you tackled them
  - A user guide explaining the steps to run the program
  - The whole source code (in Courier New with font size:8) with comments

The group leader is asked to upload everything on LMS as a compressed file (zip) containing (source code and report).

All submissions will be examined using plagiarism detection software. **Zero** will be given to any submissions with similarities.

Presentations and discussions will be conducted at the end of the semester.

Good luck and have fun!

Appendix

If you are new to python, here is an example of class Student, to help you get used to the syntax:

```
#Definition of class Student with two attributes name and id
#Notice that all methods must take "self" as first parameter in the definition

class Student:
    #similar to constructor, defines attributes
    def __init__(self, name, stid):
        self.name= name
        self.stid= stid

    def willGraduate(self):
        return self.stid[:3] == "434"

    #similar to toString()
    def __repr__(self):
        return "Student Name: "+self.name+"\nID: "+self.stid

    #similar to equals(Student other)
    def __eq__(self, other):
        if isinstance(other, Student):
            return ((self.name == other.name) and (self.stid == other.stid))
        else:
            return False
    #not equals
    def __ne__(self, other):
        return (not self.__eq__(other))

    #finds an immutable hashcode associated with object
    #used as key in sets and dictionaries
    def __hash__(self):
        return hash(self.__repr__())
```