Collaboration policy for homework: This homework is to be done by yourself. Please do not collaborate with others, look for answers online, or post homework problems or solutions to websites or discord servers. **We encourage you to come to office hours for help.**

For the code, modify the provided `hw2.py` and submit that.

Problem 1. (40 points)

For problem 1, make alterations to the given python module. Use `the example_problem` function as a guide for how to create an instance of the 8-puzzle and what should be returned.

a. In the function `problem_1a`, create an instance of the 8-puzzle with initial state (1, 6, 3, 4, 0, 8, 2, 7, 5) and run A* search on it with the default heuristic. Return the solution that A* search finds.

b. In the function `problem_1b`, create an instance of the 8-puzzle with initial state (1, 4, 2, 6, 7, 3, 0, 8, 5) and goal state (1, 2, 3, 8, 0, 4, 7, 6, 5) and run A* search on it with the default heuristic. Return the solution that A* search finds.

c. In the function `problem_1c`, create an instance of the 8-puzzle with initial state (0, 6, 2, 1, 5, 3, 4, 8, 7) and goal state (1, 2, 3, 8, 0, 4, 7, 6, 5) and run A* search on it with the default heuristic. Return the solution that A* search finds.

d. In the function `problem_1d`, repeat problem c (including with the different goal state), but write a new heuristic function that calculates the Manhattan Distance. Manhattan Distance is the sum of the orthogonal (non-diagonal) distance that each tile is away from its correct location, ignoring the fact that other tiles are in the way. Return the solution that A* search finds.

e. For this one, put your answer in your pdf. Find four initial states that have optimal solutions of lengths 18, 20, 22 and 24. For each of these, run A* search using the default heuristic and your heuristic. Use a python function or unix utility to time the length of each of the runs. In your writeup, create a table with the eight timing results. (Note: When trying to come up with initial states that meet this criteria, it is acceptable to work with other students in the class. This is the **only** part of this homework in which you may collaborate.)

check the time()
Make function call()
check the time again()
return the diff in the two times

Problem 2. (30 points)

For problem 2, make alterations to the given python module. These questions use the simplified map of Romania from the textbook. An undirected graph representation of that map can be found in the search.py module. Use `example_problem_2` function as a guide for how to create a version of the pathfinding problem and track statistics about it.

a. Create a version of the problem that will find a route from Zerind to Bucharest. Using the `InstrumentedProblem` class, create three separate instances of the problem and run Breadth-First Graph search, Depth-First Graph search and Iterative Deepening search on them. Return your result in the form of a tuple of the three Problem objects.

b. Create a version of the problem that will find a route from Oradea to Hirsova. Using the `InstrumentedProblem` class, create three separate instances of the problem and run Iterative Deepening, Uniform Cost and A* search on them. Return your result in the form of a tuple of the three Problem objects.

Problem 3. (30 points)

Consider the following modification to A* search: Instead of using $f(n) = g(n) + h(n)$ as our evaluation function in the priority queue, we will instead use $f(n) = g(n) + 2 \times h(n)$.

a. Is the modified algorithm guaranteed to find a solution with an admissible heuristic? Precisely explain your reasoning.

b. If the heuristic we use is a perfect estimate (i.e. $h(n) =$ the cost of the optimal path from node n to a solution,) what will the performance of this modified A* search be? Use Big-O notation in terms of branching factor b and optimal solution path length d. Define other terms if needed.