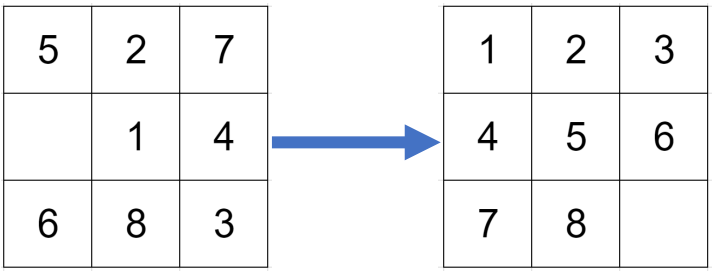
Academic honesty and plagiarism

- Plagiarism is unacceptable.
  - Code you submit must be your own. No copying, adapting, or submitting code you did not create is allowed. Working together and presenting variants of the same file is not acceptable. Here are some specific guidelines to make sure you don't cross the line:
    - Do not exchange programs or program fragments in any form on paper, via e-mail, photos, or by other means.
    - Do not copy solutions from any source, including the web or previous quarters' students.
    - Do not discuss code with other students at the level of detail that will lead to identical programs or program fragments.
    - Do not use chatGTP or other AI tools
  - Ask me if you are uncertain.
  - **All violations of the academic honesty will be immediately referred to the dean's office.**
- **Plagiarism detection will be applied to all code. It is very good and will catch you!! (20% of a class referred to dean!!)**

Lab 3 A*

3x3 Sliding Tile Puzzle

Due April 27, 7:00pm CST



For lab 3 you should use A* search to find the minimum number of actions needed to solve a given sliding tile puzzle. You are given a 9 element array that has a value 0-8 in each entry, where the value represents the tile in that location (0 represents the empty space). The locations in the array scan left to right, and then top to bottom, so for example, the state shown above on the left would be [5,2,7,0,1,4,6,8,3], and the goal state (shown above on the right) would be [1,2,3,4,5,6,7,8,0]. The only actions available to you are to move the blank tile up, right, down, or left (represented by 0,1,2,3 respectively). You should use the total (sum for all tiles, not including the empty tile) Manhattan distance between each tile location and its goal location for your admissible heuristic.

Your program should return the depth the solution was found, the number of nodes expanded (taken out of frontier), and the sequence of actions found. For example, the above puzzle should return depth=23, nodes expanded = 687, and a sequence of [1, 1, 0, 3, 3, 2, 1, 2, 3, 0, 1, 1, 2, 3, 0, 3, 2, 1, 0, 0, 1, 2, 2]

The worst-case board needs to take less than 2 min to solve, a random board should take just a few seconds.

You can only use the module copy, you cannot include any additional modules!!

Details, notes, and hints:

- The max depth for any solvable 3x3 sliding tile puzzle is 31 moves. There is one example of this given in main.py
- If there are multiple nodes in the frontier that have the same f(n), of the nodes with the lowest f(n), use the one whose single concatenated number is highest. For example if there are two states a=[5,2,7,0,1,4,6,8,3] and b=[2,5,7,0,1,4,6,8,3] with the same f(n), their single concatenated number is a=527014683 and b=257014683, so you would expand node ‘a’ first.
- You will only be given puzzles that have a reachable solution.
- When you remove a node from the frontier, also remove any other nodes in the frontier that have the same state. In other words, whenever taking a node out of the frontier, also take out any other entry from the frontier that has the same board value.

Execute and test by running: python3 main.py