# Kenneth Mead CSI-5130-13154 / CSI-4130-15496.202110

# Homework 3 A\* 15-puzzle solver

**Initial Info**

OS: Windows 10

Libraries used:

These should all be default python libraries

import sys

import time

import enum

import random

**Algorithm Analysis**

**Heuristic:** Number of misplaced tiles on the board. Calculated with Hamming distance.

Hamming distance is the number of substitutions needed to make 2 strings match.

Example of Hamming distance,

Boards tiles are stored as strings, such as

“123456789ABCDEF “

Then, take these two boards for example

“12345678 9ABCDEF“

“123456789ABCDEF “

The hamming distance here is 8

**Admissible:** This Heuristic is admissible because each board chosen will always have an equivalent or lower h-cost. Therefore, this heuristic will either eventually hit a h-cost of 0 or run until there are no possible boards left to check.

**Consistency:** Hamming distance is a stable function. Therefore, the algorithm is also stable

**Memory:** Memory is fairly efficient. I created a class called Board, and the algorithm populates 2 lists with Boards. One for visited boards and the other for boards in consideration

1 Board is approx. 64 bytes

The worst case is 15!/2 = 653,837,184,000

41,845,579,776,000 bytes

= 40,864,824,000 kilobytes

= 39,907,054 megabytes

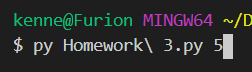
= 38,971 gigabytes

= 38 terabytes

**Run time:** Runtime is very inconsistent because of the nature of A\*. Finding the optimal solution for 1 board may be vastly different from a board that has a 1 move difference from it. This is partially derived from A\*’s tendency to visit the lowest cost states in an arbitrary order.

**Better Heuristic:** A better heuristic could be where we consider both the hamming distance of a board as a whole, and the Manhattan distance for each tile.

**How To Run**



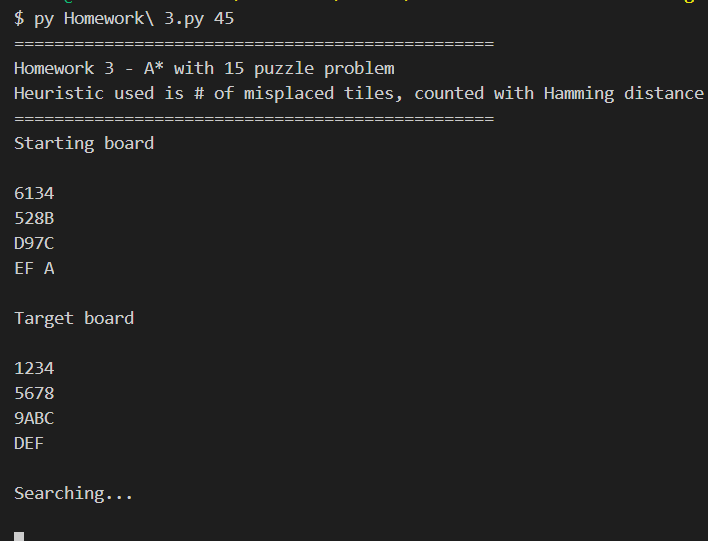
The starting board is “123456789ABCDEF “

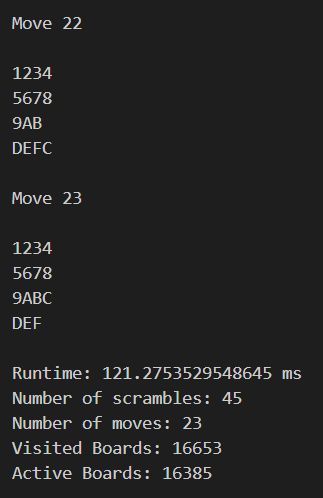
It accepts 1 optional argument which is the number of scrambles applied to the starting board. I do this so that the algorithm always handles a solvable board.

The default for the argument is 10

A good number of scrambles to use is 25

You should see output like so





Be wary of large scramble numbers (like 45). They may take a long, long time to find a solution. The algorithm does cut off when it has iterated 15!/2 times.