**COSC3P71 – Summary**

In my assignment I chose to implement the A\* heuristic algorithm. I noticed that each time I compiled my code, the time it took to find the optimal path varied by about 10 milliseconds at most. I used the heuristic function as the addition of the Manhattan distance from every tiles recent position to its end position. This ended up effecting my run time. The A\* method I implemented does not affect the number of steps in the solution. In order for the A\* algorithm to find the shortest optimal path, the heuristic cannot be bigger than the actual number of steps to reach the goal. So it is very limited in solving a randomly generated puzzle therefore I had to play around and figure out which examples will work with my algorithm.

To use my program just import into any IDE and run and it will show you each step it took to get to the optimal solution. Only works with 8 puzzle and A\* algorithm. There are examples at the bottom of the code which you can use to test the code to see if the A\* algorithm works.