Implement the 8-puzzle problem using A* algorithm, using

Heuristic function as Manhattan distance with depth not more the 3.

If goal state is not reached within this limit, agent must report

“NOSOLUTION”.

8 2 3

4 6

7 5 1

Start state

1 2 3

4 5 6

7 8

Goal State

CODE:

```python
GoalNode = [[1, 2, 3], [4, 5, 6], [7, 8, 0]]
StartNode = [[8, 2, 3], [0, 4, 6], [7, 5, 1]]
temp = []
h1 = -1
h2 = 0

print("Given StartNode is: ", StartNode)

print("\n\n\t Given GoalNode is: ", GoalNode)

print("\n\n########################################")

for i in range(len(StartNode)):
    for j in range(len(StartNode)):
        if StartNode[i][j] != GoalNode[i][j]:
            h1 += 1
print("\n\n\t h1 : Number of misplaced tiles =>", h1)

'''
for i in StartNode:
    for j in i:
        print("StartNode",j)

print("########################################")
for i in GoalNode:
    for j in i:
        print("GoalNode",j)
print("########################################")
for i in range(len(StartNode)):
```

```
    for j in range (len(StartNode)):
        print("i is ",i,"j is :",j)'''

print("\n\n#####################################")

print("\n\nDistances of the tiles from their goal positions are: \n")

for i in range(len(StartNode)):
    for j in range(len(StartNode)):
        if (StartNode[i][j] == 0):
            pass
        else:
            if (GoalNode[0][0] == StartNode[i][j]):
                temp.append(abs(i - 0) + abs(j - 0))
                print("\t", temp)

            elif (GoalNode[0][1] == StartNode[i][j]):
                temp.append(abs(i - 0) + abs(j - 1))
                print("\t", temp)
            elif (GoalNode[0][2] == StartNode[i][j]):
                temp.append(abs(i - 0) + abs(j - 2))
                print("\t", temp)
            elif (GoalNode[1][0] == StartNode[i][j]):
                temp.append(abs(i - 1) + abs(j - 0))
                print("\t", temp)
            elif (GoalNode[1][1] == StartNode[i][j]):
                temp.append(abs(i - 1) + abs(j - 1))
                print("\t", temp)
            elif (GoalNode[1][2] == StartNode[i][j]):
```

```
            print("\t", temp)
        elif (GoalNode[1][1] == StartNode[i][j]):
            temp.append(abs(i - 1) + abs(j - 1))
            print("\t", temp)
        elif (GoalNode[1][2] == StartNode[i][j]):
            temp.append(abs(i - 1) + abs(j - 2))
            print("\t", temp)
        elif (GoalNode[2][0] == StartNode[i][j]):
            temp.append(abs(i - 2) + abs(j - 0))
            print("\t", temp)
        elif (GoalNode[2][1] == StartNode[i][j]):
            temp.append(abs(i - 2) + abs(j - 1))
            print("\t", temp)
        elif (GoalNode[2][2] == StartNode[i][j]):
            temp.append(abs(i - 2) + abs(j - 2))
            print("\t", temp)
        else:
            print("Warning!!! This is for 8-puzzle program.So, don't cross the array limit.")

print("\n\n#####################################")

for i in range(len(temp)):
    h2 += temp[i]
print("\nh2 :  The sum of the distances of the tiles from their goal positions =>", h2)

h = h1 + h2

print("\n\n\tSo, the instance of given 8-puzzle solution is", h, "steps long.")
```

output:

```
C:\Users\Madhan\PycharmProjects\main\venv\Scripts\python.exe C:/Users/Madhan/PycharmProjects/main/madan.py
Given StartNode is:  [[8, 2, 3], [0, 4, 6], [7, 5, 1]]

     Given GoalNode is:  [[1, 2, 3], [4, 5, 6], [7, 8, 0]]

####################################

     h1 : Number of misplaced tiles => 4

####################################

Distances of the tiles from their goal positions are:

     [3]
     [3, 0]
     [3, 0, 0]
     [3, 0, 0, 1]
     [3, 0, 0, 1, 0]
     [3, 0, 0, 1, 0, 0]
     [3, 0, 0, 1, 0, 0, 1]
     [3, 0, 0, 1, 0, 0, 1, 4]

####################################

h2 :  The sum of the distances of the tiles from their goal positions => 9
```

So, the instance of given 8-puzzle solution is 13 steps long.