**AI FOR LLM- CSA1704**

**1. 8 PUZZLE PROBLEM**

**CODE:**

from collections import deque

# Function to print puzzle board

def print\_board(board):

for i in range(0, 9, 3):

print(board[i:i+3])

print()

# Function to get possible next states

def get\_moves(state):

moves = []

pos = state.index(0) # blank space

x, y = divmod(pos, 3)

directions = {"UP": (x-1, y), "DOWN": (x+1, y), "LEFT": (x, y-1), "RIGHT": (x, y+1)}

for move, (nx, ny) in directions.items():

if 0 <= nx < 3 and 0 <= ny < 3:

new = state[:]

new[pos], new[nx\*3+ny] = new[nx\*3+ny], new[pos]

moves.append((move, new))

return moves

# BFS Solver

def solve(start, goal):

q = deque([(start, [])])

visited = set()

while q:

state, path = q.popleft()

if state == goal:

return path

visited.add(tuple(state))

for move, new\_state in get\_moves(state):

if tuple(new\_state) not in visited:

q.append((new\_state, path+[move]))

return None

# Main

start = [1,2,3,4,0,6,7,5,8]

goal = [1,2,3,4,5,6,7,8,0]

print("Initial Board:")

print\_board(start)

solution = solve(start, goal)

if solution:

print("Solution found in", len(solution), "moves:", solution)

state = start[:]

for move in solution:

print("Move:", move)

for m, new\_state in get\_moves(state):

if m == move:

state = new\_state

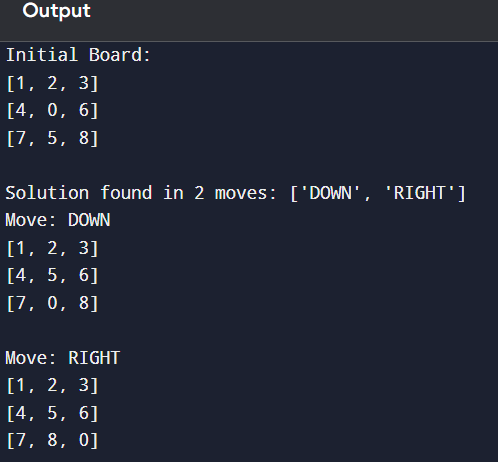
print\_board(state)

break

else:

print("No solution exists!")

**OUTPUT:**

****