1. 8 puzzle program

class Puzzle8:

def \_init\_(self, initial\_state=None):

self.goal\_state = [[1, 2, 3], [4, 5, 6], [7, 8, 0]]

if initial\_state:

self.state = initial\_state

else:

self.state = self.goal\_state.copy()

self.moves = 0

def get\_user\_input(self):

print("Enter the initial state of the puzzle row by row, use '0' for the blank tile:")

self.state = []

for \_ in range(3):

row = input().split()

self.state.append([int(cell) for cell in row])

def get\_valid\_moves(self):

row, col = self.find\_blank()

valid\_moves = []

if row > 0:

valid\_moves.append((-1, 0)) # Move the blank tile up

if row < 2:

valid\_moves.append((1, 0)) # Move the blank tile down

if col > 0:

valid\_moves.append((0, -1)) # Move the blank tile left

if col < 2:

valid\_moves.append((0, 1)) # Move the blank tile right

return valid\_moves

def move(self, direction):

row, col = self.find\_blank()

dr, dc = direction

self.state[row][col], self.state[row + dr][col + dc] = self.state[row + dr][col + dc], self.state[row][col]

self.moves += 1

def find\_blank(self):

for i, row in enumerate(self.state):

if 0 in row:

return i, row.index(0)

def \_str\_(self):

return "\n".join(" ".join(str(cell) for cell in row) for row in self.state)

# Example usage

puzzle = Puzzle8()

puzzle.get\_user\_input()

print("Entered puzzle:")

print(puzzle)

print("Valid moves:", puzzle.get\_valid\_moves())

OUTPUT:

