Date Page

8puzzle problem

IBM18CS049
MK Gagan Roshan

```
import copy, math
# Depth first search in search of
  target - using recursion.

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

def solve(src, target, limit, visited states):
    if src == target:
        print("Target is Reached in stage:", limit)
        return True
    if limit >= 12:
        return False
        visited_states.append(src)
        actions = possible_moves(src, visited states)
        new move = []
        min = math.inf
    for action in actions:
    if action not in visited_states and manhattan
    (action, target, limit) > min:
        min = manhattan(action, target, limit)
        new_move = action
        print("Stage :", limit+1)
        display(new_move)
        if solve(new_move, target, limit+1) visited_states)
          is True:
            return True
        return False
```

```
def index(mylist,v):
    for i,x in enumerate(mylist):
        if v in x:
            return (i,x.index(v))
def possible_moves(state,visited_states):
    b = index(state, 0)
    d = []
    if b[0] ≠ 2:
        d.append('d')
    if b[0]>0:
        d.append('u')
    if b[1]≠2:
        d.append('r')
    if b[1]>0:
        d.append('l')
    pos_moves = []
    for i in d:
        move=gen(state,i,b)
    if move not in visited_states:
        pos_moves.append(move)
def gen(state,m,b):
    temp = copy.deepcopy(state)
    if m=='u':
        temp[b[0]][b[1]]=temp[b[0]-1][b[1]]
    elif m=='d':
        temp[b[0]][b[1]]=temp[b[0]+1][b[1]]
                    temp[b[0]+1][b[1]]=0
    elif m=='l':
        temp[b[0]][b[1]]=temp[b[0]][b[1]-1]
                    temp[b[0]][b[1]-1]=0
    elif m='r':
        temp[b[0]][b[1]]=temp[b[0]][b[1]+1]
                    temp[b[0]][b[1]+1]=0
    if "x"=="target"
    return temp
```