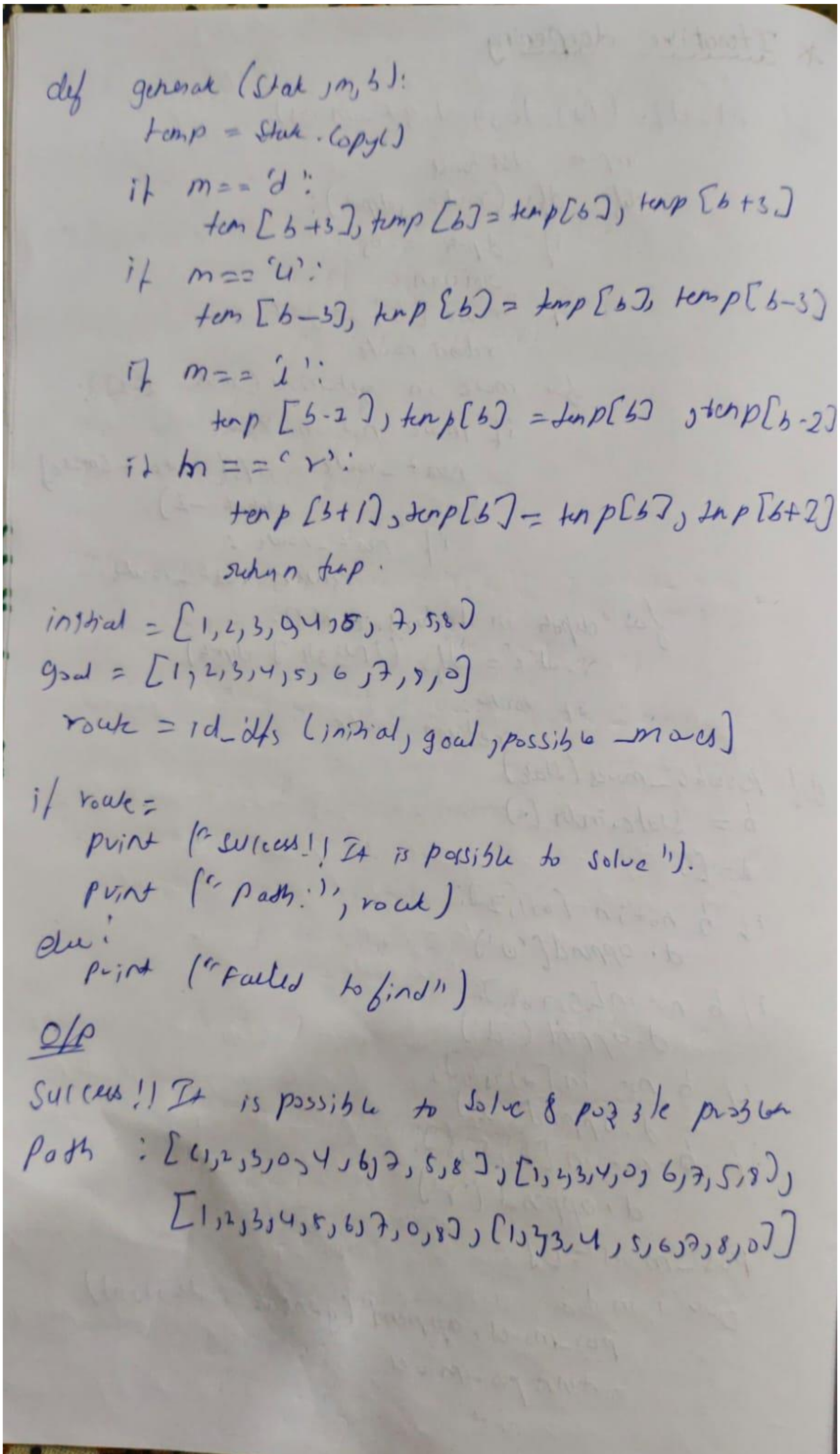
* Iterative deepening

```
def id-dfs (Pu33)e, goal, get_moves):
        import iter tools
        def dfs (route, depth):
            if depth ==0;
                return
            if route [-1] == goal:
                return route
            for move in get_moves (route [-1]):
                if move not in route:
                    next_route = dfs( route + [move]
                                  depth -1)
                    if next_route :
                        return next_route
        for depth in itertools. count ():
            route = dfs ([Pu33le ], depth)
            if route:
                return route
def Possible_moves (State)
    b = State.index (0)
    d= []
    if b not in [0,1,2]:
        d. append['u')
    if b not in [6,7,8 ]:
        d. append ('d')
    if b not in [0,3,6];
        d. append ('l')
    if b not in [2 ,5,8 ]:
        d. append ('r')
    pos_moves = []
    for i in d:
        pos_moves. append (gensate(state, i, b))
        return pos_moves.
```



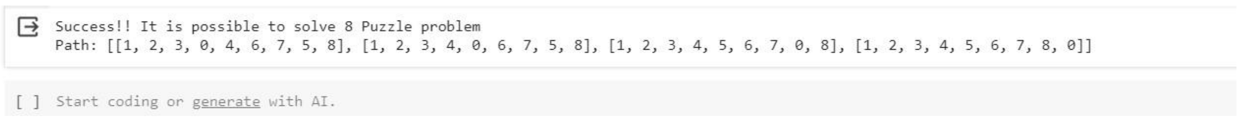
```
def generate (state ,m, b):
    temp = state .copy()
    if m== 'd':
        tem [b+3], temp [b] = temp[b], temp [b+3]
    if m== 'u':
        tem [b-3], temp [b] = temp [b], temp[b-3]
    if m== 'l':
        temp [b-1], temp[b] =temp[b] ,temp[b-2]
    if m == 'r':
        temp [b+1], temp[b] = temp[b], temp [b+2]
        return temp.
initial = [1,2,3,0,4,6, 7,5,8]
goal = [1,2,3,4,5, 6 ,7,8,0]
 route = id_dfs (initial, goal ,possible _moves]

if route=
    print ("success!! It is possible to solve").
    print ("path:", route)
else:
    print ("Failed to find")
```

<u>O/P</u>

```
Success !! It is possible to Solve 8 puzzle problem
Path : [[1,2,3,0,4,6,7,5,8 ], [1,2,3,4,0, 6,7,5,8],
        [1,2,3,4,5,6,7,0,8], [1,2,3,4 ,5,6,7,8,0]]
```

Output:



```
Success!! It is possible to solve 8 Puzzle problem
Path: [[1, 2, 3, 0, 4, 6, 7, 5, 8], [1, 2, 3, 4, 0, 6, 7, 5, 8], [1, 2, 3, 4, 5, 6, 7, 0, 8], [1, 2, 3, 4, 5, 6, 7, 8, 0]]
```