8 puzzle using BFS

```
def bfs(src,target):
    queue = []
    queue.append(src)

    exp = []

    while len(queue) > 0:
        source = queue.pop(0)
        exp.append(source)

        print(source)

        if source==target:
            print("success")
            return

        poss_moves_to_do = []
        poss_moves_to_do = possible_moves(source,exp)

        for move in poss_moves_to_do:

            if move not in exp and move not in queue:
                queue.append(move)

def possible_moves(state,visited_states):
    #index of empty spot
    b = state.index(-1)

    #directions array
    d = []
    #Add all the possible directions

    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')

# If direction is possible then add state to move
    pos_moves_it_can = []

    # for all possible directions find the state if that move is played
    #Jump to gen function to generate all possible moves in the given
directions

    for i in d:
        pos_moves_it_can.append(gen(state,i,b))

    return [move_it_can for move_it_can in pos_moves_it_can if move_it_can
not in visited_states]
```

```
def gen(state, m, b):
    temp = state.copy()

    if m=='d':
        temp[b+3],temp[b] = temp[b],temp[b+3]

    if m=='u':
        temp[b-3],temp[b] = temp[b],temp[b-3]

    if m=='l':
        temp[b-1],temp[b] = temp[b],temp[b-1]

    if m=='r':
        temp[b+1],temp[b] = temp[b],temp[b+1]


    # return new state with tested move to later check if "src == target"
    return temp


src = [1,2,3,-1,4,5,6,7,8]
target = [1,2,3,4,5,-1,6,7,8]
bfs(src, target)
```

OUTPUT-

```
[1, 2, 3, -1, 4, 5, 6, 7, 8]
[1, 2, 3, 6, 4, 5, -1, 7, 8]
[-1, 2, 3, 1, 4, 5, 6, 7, 8]
[1, 2, 3, 4, -1, 5, 6, 7, 8]
[1, 2, 3, 6, 4, 5, 7, -1, 8]
[2, -1, 3, 1, 4, 5, 6, 7, 8]
[1, 2, 3, 4, 7, 5, 6, -1, 8]
[1, -1, 3, 4, 2, 5, 6, 7, 8]
[1, 2, 3, 4, 5, -1, 6, 7, 8]
success
```