8 PUZZLE using IDDFS

```
def dfs(src,target,limit,visited_states):
    if src == target:
        return True
    if limit <= 0:
        return False
    visited_states.append(src)
    moves = possible_moves(src,visited_states)
    for move in moves:
        if dfs(move, target, limit-1, visited_states):
            return True
    return False



def possible_moves(state,visited_states):
    b = state.index(-1)
    d = []
    if b not in [0,1,2]:
        d += 'u'
    if b not in [6,7,8]:
        d += 'd'
    if b not in [2,5,8]:
        d += 'r'
    if b not in [0,3,6]:
        d += 'l'
    pos_moves = []
    for move in d:
        pos_moves.append(gen(state,move,b))
    return [move for move in pos_moves if move not in visited_states]



def gen(state, move, blank):
    temp = state.copy()
    if move == 'u':
        temp[blank-3], temp[blank] = temp[blank], temp[blank-3]
    if move == 'd':
        temp[blank+3], temp[blank] = temp[blank], temp[blank+3]
    if move == 'r':
        temp[blank+1], temp[blank] = temp[blank], temp[blank+1]
    if move == 'l':
        temp[blank-1], temp[blank] = temp[blank], temp[blank-1]
    return temp


def iddfs(src,target,depth):
    for i in range(depth):
        visited_states = []
        if dfs(src,target,i+1,visited_states):
            return True
    return False
```

OUTPUT

Test 1

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

```
False
```

Test 3

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

```
True
```

Test 4

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

for i in range(1, 100):
    val = iddfs(src,target,i)
    print(i, val)
    if val == True:
        break
```

```
1 False
2 False
3 False
4 False
5 False
6 False
7 False
8 False
9 False
10 False
11 False
12 False
13 False
14 False
15 False
16 False
17 False
18 False
19 False
20 False
21 False
22 False
23 False
24 False
25 True
```