M.k Gogom Rodhan
1BM18CS041

①

```
def print_grid(src):
    state = src.copy()
    state[state.index(-1)] = ' '
    print(
        f"""
        {state[0]}{state[1]}{state[2]}
        {state[3]}{state[4]}{state[5]}
        {state[6]}{state[7]}{state[8]}
        """)

def h(state, target):
    dist = 0
    for i in state:
        d1, d2 = state.index(i),
    target.index(i)
        x1, y1 = d1 % 3, d1 // 3
        x2, y2 = d2 % 3, d2 // 3
    dist += abs(x1-x2) + abs(y1-y2)
    return dist
                        depth
def astar(src, target):
    states = [src]
    g = 0
    visited_states = set()
    while len(states):
        print(f"Level: {g}")
        moves = []
        for state in states:
            visited_states.add(tuple(state))
            print_grid(state)
            if state == target:
```


Gogom

```
print("Success")
return
moves = [move for move in
possible_moves(state, visited_states) if
move not in moves]
    #print(moves)
    costs = [g+h(move,target) for
move in moves]
  #print(costs)
   states =[moves[i]
        for i in range(len(moves)) if
costs[i]==min(costs)]
    g+=1
  print("Fail")
def possible_moves(state,visited_states):
    b=state.index(-1)
    d=[]
    if 9>b-3>=0:
      d+='u'
    if 9-b+3>=0:
      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 tuple
    (move) not in visited_states]
```


Gingas

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

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

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

depth = 3
astar(src, target, depth)
```