Dhanush Pyla
AP21110010849
CSE-N

AIML LAB WEEK – 6

```
import heapq


GOAL_STATE = [

   [1, 2, 3],

   [4, 5, 6],

   [7, 8, 0]

]

MOVES = [(-1, 0), (1, 0), (0, -1), (0, 1)]


def is_valid_move(x, y):

   return 0 <= x < 3 and 0 <= y < 3


def calculate_heuristic(state):

   misplaced_tiles = 0

   for i in range(3):

       for j in range(3):

           if state[i][j] != GOAL_STATE[i][j]:

               misplaced_tiles += 1

   return misplaced_tiles


def calculate_cost(state, level):

   return level


def create(initial_state, max_level):

   initial_state = [list(row) for row in initial_state]

   priority_queue = [(calculate_cost(initial_state, 0) + calculate_heuristic(initial_state), 0,
initial_state)]


   while priority_queue:
       _, current_level, current_state = heapq.heappop(priority_queue)

       if current_state == GOAL_STATE:
           print("Goal State Reached!")
           return

       if current_level > max_level:
           continue

       print("Level:", current_level)
       for row in current_state:
           print(" ".join(map(str, row)))
       print("Heuristic Value (Level + Misplaced Tiles):", current_level +
calculate_heuristic(current_state))
       print()

       for i in range(3):
           for j in range(3):
               if current_state[i][j] == 0:
                   empty_x, empty_y = i, j
       for dx, dy in MOVES:
           new_x, new_y = empty_x + dx, empty_y + dy
           if is_valid_move(new_x, new_y):
               new_state = [list(row) for row in current_state]
               new_state[empty_x][empty_y], new_state[new_x][new_y] = new_state[new_x][new_y],
new_state[empty_x][empty_y]
               new_level = current_level + 1
               priority = new_level + calculate_heuristic(new_state)  # f-value
               heapq.heappush(priority_queue, (priority, new_level, new_state))

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

level_number = int(input("Enter the max level required for the state space tree to be produced: "))
create(initial_state, level_number)
```