As you all know about 8-puzzle game discussed in the AI class.
You must write a function called *Create (Initial state, Level number)* to generate all the states of 8-puzzle game starting from the given initial state up to a particular level. The initial state and level number are the input to the *Create* function. Use the suitable data structure to represent the state of the 8-puzzle game and use the actions (Up, Down, Left, and Right) to generate the new states. After creating each state of the problem calculate the heuristic value of that state. The heuristic function h(S) = number of tiles misplaced with respect to the goal state (you can define your own goal state or use the goal state which was discussed in the AI class). Print all the states which are generated by the *Create* function with their heuristic values.

CODE:

```
import copy

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

def 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 generate_new_states(current_state):
    new_states = []
    zero_row, zero_col = None, None

    for i in range(3):
        for j in range(3):
            if current_state[i][j] == 0:
                zero_row, zero_col = i, j
                break

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

    for dr, dc in moves:
        new_row, new_col = zero_row + dr, zero_col + dc

        if 0 <= new_row < 3 and 0 <= new_col < 3:
            new_state = copy.deepcopy(current_state)
```

```
            new_state[zero_row][zero_col], new_state[new_row][new_col] = new_state[new_row][new_col],
new_state[zero_row][zero_col]
            new_states.append(new_state)

    return new_states

def Create(initial_state, level):
    states_queue = [(initial_state, 0)]
    visited = set()

    while states_queue:
        current_state, current_level = states_queue.pop(0)

        if current_level > level:
            break

        print(f"Level {current_level}:\n", current_state)
        print("Heuristic Value:", heuristic(current_state))
        print()

        visited.add(tuple(map(tuple, current_state)))
        new_states = generate_new_states(current_state)

        for new_state in new_states:
            if tuple(map(tuple, new_state)) not in visited:
                states_queue.append((new_state, current_level + 1))
                visited.add(tuple(map(tuple, new_state)))

if __name__ == "__main__":
    initial_state = [[1, 2, 3], [0, 4, 6], [7, 5, 8]]
    level = 2

    Create(initial_state, level)
```

OUTPUT:

input

```
Level 0:
 [[1, 2, 3], [0, 4, 6], [7, 5, 8]]
Heuristic Value: 4

Level 1:
 [[1, 2, 3], [4, 0, 6], [7, 5, 8]]
Heuristic Value: 3

Level 1:
 [[1, 2, 3], [7, 4, 6], [0, 5, 8]]
Heuristic Value: 5

Level 1:
 [[0, 2, 3], [1, 4, 6], [7, 5, 8]]
Heuristic Value: 5

Level 2:
 [[1, 2, 3], [4, 6, 0], [7, 5, 8]]
Heuristic Value: 4

Level 2:
 [[1, 2, 3], [4, 5, 6], [7, 0, 8]]
Heuristic Value: 2
```

```
Level 2:
 [[1, 0, 3], [4, 2, 6], [7, 5, 8]]
Heuristic Value: 4

Level 2:
 [[1, 2, 3], [7, 4, 6], [5, 0, 8]]
Heuristic Value: 5

Level 2:
 [[2, 0, 3], [1, 4, 6], [7, 5, 8]]
Heuristic Value: 6
```