08/12/23

Date __/__/__

Page _____

LAB-4

8 puzzle A* Approach

```
Class Node:
    def (self, state, parent = None, g = 0, h = 0)
        self.state = state
        self.parent = parent
        self.g = g
        self.h = h

    def f(self):
        return self.g + self.h

def astar (start, goal, heuristic):
    openset = [Node (start, None, 0, heuristic (start))]
    visited_set = set()

    while openset:
        current_node = heapq.heappop(openset)

        if current_node.state == goal:
            return reconstruct_path (current_node)

        visitedset.add (current_node.state)
        successors = expand_node (current_node)

        for successor in successors:
            if successor.state in visitedset:
                continue

            if successor.state not in (node.state
                                       for node in open_set)
                heapq.heappush (open_set, successor)
```

```
    else:
        existing_node = next(
            node for node in open_set if
            node.state == successor.
        if successor.g < existing_node.g:
            existing_node.parent = successor.parent
            existing_node.g = successor.g

    return None.

def expand_node(node):
    moves = [(-1, 0), (1, 0), (0, -1), (0, 1)]

    for move in moves:
        new_state = (node.state[0] + move[0],
                     node.state[1] + move[1])

        if 0 <= new_state[0] < len(grid) and
            0 <= new_state[1] < len(grid)

            if grid[new_state[0]][new_state[1]]
                != "obstacle":
                new_g = node.g + 1

                new_h = heuristic_function(new_state)

                successor = Node(new_state, node,
                                 new_g, new_h)
                successors.append(successor)
    return successor
```

Date __/__/__

Page _____

```
def reconstruct_path(node):
    path = []
    while node:
        path.append(node.state)
        node = node.parent
    return path[::-1]
```

```
Harshitha R-1BM21CS075
Enter the start state matrix

1 2 3
4 5 6
_ 7 8
Enter the goal state matrix

1 2 3
4 5 6
7 8 _




  |
  |
 \'/

1 2 3
4 5 6
_ 7 8

  |
  |
 \'/

1 2 3
4 5 6
7 _ 8

  |
  |
 \'/

1 2 3
4 5 6
7 8 _
```