Exp. No: 4

# $A^*$ Search.

Aim:

To find the shortest path from a start node to the goal node using $A^*$ search algorithm.

```
def a_star(grid, start, goal):

    def heuristic(a, b):
        return abs(a[0] - b[0]) + abs(a[1] - b[1])

    rows, cols = len(grid), len(grid[0])
    open_list = [(0 + heuristic(start, goal), 0, start)]
    came_from = {}
    cost_so_far = {start: 0}
    while open_list:
        current_cost, current = heappop(open_list)
        if current == goal:
            path = []
            while current in came_from:
                path.append(current)
                current = came_from[current]
            path.append(start)
            return path[::-1]
        for dx, dy in [(-1, 0), (1, 0), (0, -1), (0, 1)]:
            neighbour = (current[0] + dx, current[1] + dy)
```

if (0 <= neighbour[0] < rows and 0 <= neighbour[1] < cols and

grid[neighbour[0]][neighbour[1]] == 0):

new_cost = cost_so_far[current] + 1

if neighbour not in cost_so_far or new_cost < cost_so_far[neighbour]:

cost_so_far[neighbour] = new_cost

priority = new_cost + heuristic(goal, neighbour)

heappush(open_list, (priority, new_cost, neighbour))

came_from[neighbour] = current.

Result:

The program is successfully executed and o/p is executed