```
Total time: 8.433s
File: /Users/rishabhjain/Documents/Masters/SEM 2/Aritificial Intelligence/Program/Program 1/program_1.py
Function: expand at line 48

Line #      Hits         Time      Per Hit    % Time    Line Contents
==============================================================
    48                                                  @cpu
    49                                                  def expand(board):
    50    232876       99.9ms            .      1.2%        for i in range(len(board.data)):
    51    582167      169.0ms            .      2.0%            for j in range(len(board.data[i])):
    52    465729      143.8ms            .      1.7%                if board.data[i][j] == '*':
    53     58219       16.2ms            .      0.2%                    location = [i,j];
    54     58219       12.7ms            .      0.2%                    break
    55
    56     58219       12.6ms            .      0.1%        actions = []
    57    214012      937.0ms            .     11.1%        for move in possible_actions(constants.board, location):
    58    155793        7.03s            .     83.3%            actions.append([result(location, move, board.data) , move])
    59
    60     58219       13.5ms            .      0.2%        return actions


Total time: 0.440s
File: /Users/rishabhjain/Documents/Masters/SEM 2/Aritificial Intelligence/Program/Program 1/program_1.py
Function: possible_actions at line 62

Line #      Hits         Time      Per Hit    % Time    Line Contents
==============================================================
    62                                                  @cpu
    63                                                  def possible_actions(board, location):
    64     58219       20.6ms            .      4.7%        actions = ["RIGHT","LEFT","UP","DOWN"]
    65     58219       14.8ms            .      3.4%        actionstopeform = []
    66
    67    291095       74.4ms            .     16.9%        for x in actions:
    68                                                          # for moving right
    69    232876       60.6ms            .     13.8%            if x == "RIGHT":
    70     58219       21.4ms            .      4.9%                if location[1]+1 < len(board):
    71     39012       29.6ms            .      6.7%                    actionstopeform.append([x,location[0],location[1]+1])
    72                                                          # for moving left
    73    174657       46.9ms            .     10.7%            elif x == "LEFT":
    74     58219       19.1ms            .      4.3%                if location[1]-1 >= 0:
    75     38989       23.4ms            .      5.3%                    actionstopeform.append([x,location[0],location[1]-1])
    76                                                          # for moving up
    77    116438       29.5ms            .      6.7%            elif x == "UP":
    78     58219       18.9ms            .      4.3%                if location[0]-1 >= 0:
    79     38894       16.0ms            .      3.6%                    actionstopeform.append([x,location[0]-1,location[1]])
    80                                                          # for moving down
    81     58219       14.4ms            .      3.3%            elif x == "DOWN":
    82     58219       19.7ms            .      4.5%                if location[0]+1 < len(board):
    83     38898       16.6ms            .      3.8%                    actionstopeform.append([x,location[0]+1,location[1]])
    84
    85     58219       13.9ms            .      3.2%        return actionstopeform


Total time: 6.632s
File: /Users/rishabhjain/Documents/Masters/SEM 2/Aritificial Intelligence/Program/Program 1/program_1.py
Function: result at line 87

Line #      Hits         Time      Per Hit    % Time    Line Contents
==============================================================
    87                                                  @cpu
    88                                                  def result(location,action,board):
    89                                                      # copy of a board so that we can modify it
    90    155793        5.48s            .     82.6%        newBoard = copy.deepcopy(board)
    91    155793      384.1ms            .      5.8%        temp = copy.deepcopy(newBoard[action[1]][action[2]])
    92    155793      371.9ms            .      5.6%        newBoard[action[1]][action[2]] = copy.deepcopy('*')
    93    155793      362.5ms            .      5.5%        newBoard[location[0]][location[1]]  = copy.deepcopy(temp)
    94                                                      # return new board after moving * - NIL to the new location
    95    155793       33.2ms            .      0.5%        return newBoard


Total time: 190.421s
File: /Users/rishabhjain/Documents/Masters/SEM 2/Aritificial Intelligence/Program/Program 1/program_1.py
Function: bfs at line 139

Line #      Hits         Time      Per Hit    % Time    Line Contents
==============================================================
   139                                                  @cpu
   140                                                  def bfs(board):
   141         1            .            .         .        frontier = queue.Queue()
   142         1            .            .         .        node = Node(data = board)
   143         1            .            .         .        frontier.put(node)
   144         1            .            .         .        maxQueueSize = 1
   145         1            .            .         .        if constants.goalBoard == node.data:
   146                                                          print('mill gya:', maxQueueSize)
   147                                                          return node
   148
   149         1            .            .         .        reached = []
   150         1            .            .         .        reached.append(board)
   151
   152     58219      194.7ms            .      0.1%        while not frontier.empty():
   153     58219      374.5ms            .      0.2%            val = frontier.get()
   154    214010        9.12s            .      4.8%            for child in expand(val):
   155    155792      414.2ms            .      0.2%                s =  Node(data=child[0], depth = val.depth + 1, move= child[1] , prev=val)
   156
   157    155792       97.2ms            .      0.1%                if goalBoard == s.data:
   158         1        0.1ms        0.1ms         .                    print('Max queue size:', maxQueueSize)
   159         1            .            .         .                    return s
   160    155791      179.20s        1.2ms     94.1%                if s.data not in reached:
```

```
   161     76968      122.1ms            .      0.1%                    reached.append(s.data)
   162     76968      872.1ms            .      0.5%                    frontier.put(s)
   163     76968       32.1ms            .         .                    maxQueueSize+=1
   164
   165                                                          print('Max queue size:', maxQueueSize)
   166
   167                                                          return failure


Total time: 0.001s
File: /Users/rishabhjain/Documents/Masters/SEM 2/Aritificial Intelligence/Program/Program 1/program_1.py
Function: printStatistics at line 258

Line #      Hits         Time      Per Hit    % Time    Line Contents
==============================================================
   258                                                  @cpu
   259                                                  def printStatistics(solution):
   260         1            .            .      0.7%        pathCost = 0
   261         1            .            .      0.2%        stateSequence = []
   262         1            .            .         .        actionSequence = []
   263
   264        23            .            .      3.1%        while solution.prev != None:
   265        22            .            .      2.2%            stateSequence.insert(0, solution.data)
   266        22            .            .      2.0%            actionSequence.insert(0, solution.move)
   267        22            .            .      1.1%            solution = solution.prev
   268        22            .            .      1.3%            pathCost += 1
   269
   270         1            .            .      8.6%        print('Action sequence:')
   271         1        0.1ms        0.1ms     18.0%        print(*actionSequence, sep='\n')
   272
   273         1            .            .      0.5%        print('\nState sequence:')
   274         1        0.3ms        0.3ms     53.2%        print(*stateSequence, sep='\n')
   275
   276         1        0.1ms        0.1ms      9.2%        print('\nPath cost:', pathCost)
```