```
Total time: 7.127s
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    201468       85.6ms            .      1.2%        for i in range(len(board.data)):
    51    503660      137.5ms            .      1.9%            for j in range(len(board.data[i])):
    52    402926      120.9ms            .      1.7%                if board.data[i][j] == '*':
    53     50367       13.5ms            .      0.2%                    location = [i,j];
    54     50367       10.5ms            .      0.1%                    break
    55
    56     50367       10.3ms            .      0.1%        actions = []
    57    186814      767.2ms            .     10.8%        for move in possible_actions(constants.board, location):
    58    136447        5.97s            .     83.8%            actions.append([result(location, move, board.data) , move])
    59
    60     50367       11.2ms            .      0.2%        return actions


Total time: 0.354s
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     50367       16.9ms            .      4.8%        actions = ["RIGHT","LEFT","UP","DOWN"]
    65     50367       12.3ms            .      3.5%        actionstopeform = []
    66
    67    251835       62.4ms            .     17.6%        for x in actions:
    68                                                          # for moving right
    69    201468       48.9ms            .     13.8%            if x == "RIGHT":
    70     50367       19.4ms            .      5.5%                if location[1]+1 < len(board):
    71     34145       17.1ms            .      4.8%                    actionstopeform.append([x,location[0],location[1]+1])
    72                                                          # for moving left
    73    151101       38.4ms            .     10.9%            elif x == "LEFT":
    74     50367       15.8ms            .      4.5%                if location[1]-1 >= 0:
    75     34135       15.2ms            .      4.3%                    actionstopeform.append([x,location[0],location[1]-1])
    76                                                          # for moving up
    77    100734       24.1ms            .      6.8%            elif x == "UP":
    78     50367       16.0ms            .      4.5%                if location[0]-1 >= 0:
    79     34084       13.8ms            .      3.9%                    actionstopeform.append([x,location[0]-1,location[1]])
    80                                                          # for moving down
    81     50367       12.1ms            .      3.4%            elif x == "DOWN":
    82     50367       16.6ms            .      4.7%                if location[0]+1 < len(board):
    83     34083       13.6ms            .      3.9%                    actionstopeform.append([x,location[0]+1,location[1]])
    84
    85     50367       11.2ms            .      3.2%        return actionstopeform


Total time: 5.647s
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    136447        4.67s            .     82.7%        newBoard = copy.deepcopy(board)
    91    136447      324.2ms            .      5.7%        temp = copy.deepcopy(newBoard[action[1]][action[2]])
    92    136447      313.9ms            .      5.6%        newBoard[action[1]][action[2]] = copy.deepcopy('*')
    93    136447      308.2ms            .      5.5%        newBoard[location[0]][location[1]]  = copy.deepcopy(temp)
    94                                                      # return new board after moving * - NIL to the new location
    95    136447       29.1ms            .      0.5%        return newBoard


Total time: 145.606s
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     50367      153.3ms            .      0.1%        while not frontier.empty():
   153     50367      298.7ms            .      0.2%            val = frontier.get()
   154    186812        7.68s            .      5.3%            for child in expand(val):
   155    136446      355.6ms            .      0.2%                s =  Node(data=child[0], depth = val.depth + 1, move= child[1] , prev=val)
   156
   157    136446       85.2ms            .      0.1%                if goalBoard == s.data:
   158         1        0.1ms        0.1ms         .                    print('Max queue size:', maxQueueSize)
   159         1            .            .         .                    return s
   160    136445      136.22s        1.0ms     93.6%                if s.data not in reached:
```

```
   161     68948       98.7ms            .      0.1%                    reached.append(s.data)
   162     68948      685.4ms            .      0.5%                    frontier.put(s)
   163     68948       27.6ms            .         .                    maxQueueSize+=1
   164
   165                                                          print('Max queue size:', maxQueueSize)
   166
   167                                                          return failure


Total time: 0.000s
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            .            .      0.6%        pathCost = 0
   261         1            .            .         .        stateSequence = []
   262         1            .            .         .        actionSequence = []
   263
   264        22            .            .      5.2%        while solution.prev != None:
   265        21            .            .      7.0%            stateSequence.insert(0, solution.data)
   266        21            .            .      5.8%            actionSequence.insert(0, solution.move)
   267        21            .            .      4.1%            solution = solution.prev
   268        21            .            .      4.1%            pathCost += 1
   269
   270         1            .            .     12.2%        print('Action sequence:')
   271         1        0.1ms        0.1ms     30.8%        print(*actionSequence, sep='\n')
   272
   273         1            .            .      1.2%        print('\nState sequence:')
   274         1            .            .     26.7%        print(*stateSequence, sep='\n')
   275
   276         1            .            .      2.3%        print('\nPath cost:', pathCost)
```