```
Total time: 16.092s
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    449564      202.4ms            .      1.3%        for i in range(len(board.data)):
    51   1123859      314.7ms            .      2.0%            for j in range(len(board.data[i])):
    52    899077      265.2ms            .      1.6%                if board.data[i][j] == '*':
    53    112391       34.5ms            .      0.2%                    location = [i,j];
    54    112391       23.2ms            .      0.1%                    break
    55
    56    112391       23.9ms            .      0.1%        actions = []
    57    416708        1.73s            .     10.7%        for move in possible_actions(constants.board, location):
    58    304317       13.48s            .     83.7%            actions.append([result(location, move, board.data) , move])
    59
    60    112391       24.8ms            .      0.2%        return actions


Total time: 0.805s
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    112391       39.0ms            .      4.8%        actions = ["RIGHT","LEFT","UP","DOWN"]
    65    112391       28.2ms            .      3.5%        actionstopeform = []
    66
    67    561955      140.9ms            .     17.5%        for x in actions:
    68                                                          # for moving right
    69    449564      113.0ms            .     14.0%            if x == "RIGHT":
    70    112391       41.6ms            .      5.2%                if location[1]+1 < len(board):
    71     76061       40.6ms            .      5.0%                    actionstopeform.append([x,location[0],location[1]+1])
    72                                                          # for moving left
    73    337173       90.5ms            .     11.2%            elif x == "LEFT":
    74    112391       35.6ms            .      4.4%                if location[1]-1 >= 0:
    75     76010       35.8ms            .      4.5%                    actionstopeform.append([x,location[0],location[1]-1])
    76                                                          # for moving up
    77    224782       54.4ms            .      6.8%            elif x == "UP":
    78    112391       35.0ms            .      4.3%                if location[0]-1 >= 0:
    79     76063       30.4ms            .      3.8%                    actionstopeform.append([x,location[0]-1,location[1]])
    80                                                          # for moving down
    81    112391       26.8ms            .      3.3%            elif x == "DOWN":
    82    112391       36.6ms            .      4.5%                if location[0]+1 < len(board):
    83     76183       30.8ms            .      3.8%                    actionstopeform.append([x,location[0]+1,location[1]])
    84
    85    112391       25.7ms            .      3.2%        return actionstopeform


Total time: 12.736s
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    304317       10.54s            .     82.8%        newBoard = copy.deepcopy(board)
    91    304317      731.4ms            .      5.7%        temp = copy.deepcopy(newBoard[action[1]][action[2]])
    92    304317      707.2ms            .      5.6%        newBoard[action[1]][action[2]] = copy.deepcopy('*')
    93    304317      693.5ms            .      5.4%        newBoard[location[0]][location[1]]  = copy.deepcopy(temp)
    94                                                      # return new board after moving * - NIL to the new location
    95    304317       64.1ms            .      0.5%        return newBoard


Total time: 651.005s
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    112391      412.8ms            .      0.1%        while not frontier.empty():
   153    112391      753.2ms            .      0.1%            val = frontier.get()
   154    416706       17.44s            .      2.7%            for child in expand(val):
   155    304316      896.6ms            .      0.1%                s =  Node(data=child[0], depth = val.depth + 1, move= child[1] , prev=val)
   156
   157    304316      234.2ms            .         .                if goalBoard == s.data:
   158         1        0.1ms        0.1ms         .                    print('Max queue size:', maxQueueSize)
   159         1            .            .         .                    return s
   160    304315      629.45s        2.1ms     96.7%                if s.data not in reached:
```

```
   161    136150      229.0ms            .         .                    reached.append(s.data)
   162    136150        1.53s            .      0.2%                    frontier.put(s)
   163    136150       57.6ms            .         .                    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            .            .      0.3%        stateSequence = []
   262         1            .            .         .        actionSequence = []
   263
   264        25            .            .      3.6%        while solution.prev != None:
   265        24            .            .      2.1%            stateSequence.insert(0, solution.data)
   266        24            .            .      3.3%            actionSequence.insert(0, solution.move)
   267        24            .            .      1.8%            solution = solution.prev
   268        24            .            .      2.1%            pathCost += 1
   269
   270         1            .            .      9.9%        print('Action sequence:')
   271         1        0.1ms        0.1ms     19.2%        print(*actionSequence, sep='\n')
   272
   273         1            .            .      0.9%        print('\nState sequence:')
   274         1        0.2ms        0.2ms     54.5%        print(*stateSequence, sep='\n')
   275
   276         1            .            .      1.8%        print('\nPath cost:', pathCost)
```