Artificial Intelligence

Lab 1

8 Puzzle Problem

Aman Kalla

RA1911003010640

Algorithm:-

* Define a function find\_next() . // accept node
* moves := a map defining moves as a list corresponding to each value {0: [1, 3],1: [0, 2, 4],2: [1, 5],3: [0, 4, 6],4: [1, 3, 5, 7],5: [2, 4, 8],6: [3, 7],7: [4, 6, 8],8: [5, 7],}
* results := a new list
* pos\_0 := first value of node
* for each move in moves[pos\_0], do
  + new\_node := a new list from node
  + swap new\_node[move] and new\_node[pos\_0]
  + insert a new tuple from new\_node at the end of results
* return results
* Define a function get\_paths() . This will take dict
* cnt := 0
* Do the following infinitely, do
  + current\_nodes := a list where value is same as cnt
  + if size of current\_nodes is same as 0, then
    - return -1
  + for each node in current\_nodes, do
    - next\_moves := find\_next(node)
    - for each move in next\_moves, do
      * if move is not present in dict, then
        + dict[move] := cnt + 1
      * if move is same as (0, 1, 2, 3, 4, 5, 6, 7, 8) , then
        + return cnt + 1
      * cnt := cnt + 1
* From the main method do the following:
* dict := a new map, flatten := a new list
* for i in range 0 to row count of board, do
  + flatten := flatten + board[i]
* flatten := a copy of flatten
* dict[flatten] := 0
* if flatten is same as (0, 1, 2, 3, 4, 5, 6, 7, 8) , then
  + return 0
* return get\_paths(dict)

Code:-

class Solution:

def solve(self, board):

dict = {}

flatten = []

for i in range(len(board)):

flatten += board[i]

flatten = tuple(flatten)

dict[flatten] = 0

if flatten == (0, 1, 2, 3, 4, 5, 6, 7, 8):

return 0

return self.get\_paths(dict)

def get\_paths(self, dict):

cnt = 0

while True:

current\_nodes = [x for x in dict if dict[x] == cnt]

if len(current\_nodes) == 0:

return -1

for node in current\_nodes:

next\_moves = self.find\_next(node)

for move in next\_moves:

if move not in dict:

dict[move] = cnt + 1

if move == (0, 1, 2, 3, 4, 5, 6, 7, 8):

return cnt + 1

cnt += 1

def find\_next(self, node):

moves = {

0: [1, 3],

1: [0, 2, 4],

2: [1, 5],

3: [0, 4, 6],

4: [1, 3, 5, 7],

5: [2, 4, 8],

6: [3, 7],

7: [4, 6, 8],

8: [5, 7],

}

results = []

pos\_0 = node.index(0)

for move in moves[pos\_0]:

new\_node = list(node)

new\_node[move], new\_node[pos\_0] = new\_node[pos\_0], new\_node[move]

results.append(tuple(new\_node))

return results

ob = Solution()

matrix = [

[3, 1, 2],

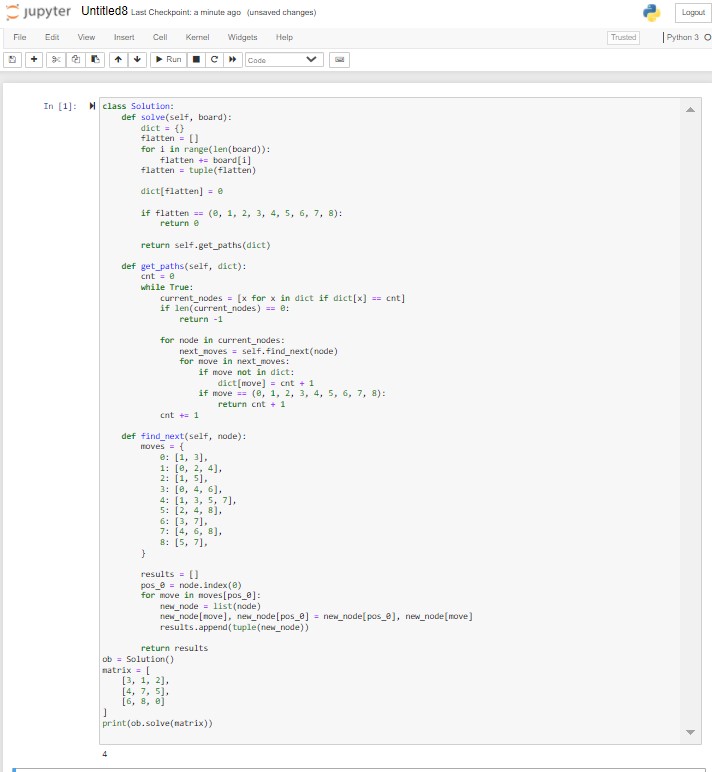
[4, 7, 5],

[6, 8, 0]

]

print(ob.solve(matrix))

Output:-



Result:- Hence the implementation of 8 Puzzle Problem is successfully executed.