program 1

tic tac toe

import os

import time

board = [' ',' ',' ',' ',' ',' ',' ',' ',' ',' ']

player = 1

########win Flags##########

Win = 1

Draw = -1

Running = 0

Stop = 1

###########################

Game = Running

Mark = 'X'

#This Function Draws Game Board

def DrawBoard():

print(" %c | %c | %c " % (board[1],board[2],board[3]))

print("\_\_\_|\_\_\_|\_\_\_")

print(" %c | %c | %c " % (board[4],board[5],board[6]))

print("\_\_\_|\_\_\_|\_\_\_")

print(" %c | %c | %c " % (board[7],board[8],board[9]))

print(" | | ")

#This Function Checks position is empty or not

def CheckPosition(x):

if(board[x] == ' '):

return True

else:

return False

#This Function Checks player has won or not

def CheckWin():

global Game

#Horizontal winning condition

if(board[1] == board[2] and board[2] == board[3] and board[1] != ' '):

Game = Win

elif(board[4] == board[5] and board[5] == board[6] and board[4] != ' '):

Game = Win

elif(board[7] == board[8] and board[8] == board[9] and board[7] != ' '):

Game = Win

#Vertical Winning Condition

elif(board[1] == board[4] and board[4] == board[7] and board[1] != ' '):

Game = Win

elif(board[2] == board[5] and board[5] == board[8] and board[2] != ' '):

Game = Win

elif(board[3] == board[6] and board[6] == board[9] and board[3] != ' '):

Game=Win

#Diagonal Winning Condition

elif(board[1] == board[5] and board[5] == board[9] and board[5] != ' '):

Game = Win

elif(board[3] == board[5] and board[5] == board[7] and board[5] != ' '):

Game=Win

#Match Tie or Draw Condition

elif(board[1]!=' ' and board[2]!=' ' and board[3]!=' ' and board[4]!=' ' and board[5]!=' ' and board[6]!=' ' and board[7]!=' ' and board[8]!=' ' and board[9]!=' '):

Game=Draw

else:

Game=Running

print("Tic-Tac-Toe Game Designed By Sourabh Somani")

print("Player 1 [X] --- Player 2 [O]\n")

print()

print()

print("Please Wait...")

time.sleep(3)

while(Game == Running):

os.system('cls')

DrawBoard()

if(player % 2 != 0):

print("Player 1's chance")

Mark = 'X'

else:

print("Player 2's chance")

Mark = 'O'

choice = int(input("Enter the position between [1-9] where you want to mark : "))

if(CheckPosition(choice)):

board[choice] = Mark

player+=1

CheckWin()

os.system('cls')

DrawBoard()

if(Game==Draw):

print("Game Draw")

elif(Game==Win):

player-=1

if(player%2!=0):

print("Player 1 Won")

else:

print("Player 2 Won")

output:-

O | O | X

\_\_\_|\_\_\_|\_\_\_

X | | O

\_\_\_|\_\_\_|\_\_\_

| | X

| |

Player 1's chance

Program 2

8 Puzzle using IDS

import copy

inp=[[1,2,3],[4,-1,5],[6,7,8]]

out=[[1,2,3],[6,4,5],[-1,7,8]]

def move(temp, movement):

if movement=="up":

for i in range(3):

for j in range(3):

if(temp[i][j]==-1):

if i!=0:

temp[i][j]=temp[i-1][j]

temp[i-1][j]=-1

return temp

if movement=="down":

for i in range(3):

for j in range(3):

if(temp[i][j]==-1):

if i!=2:

temp[i][j]=temp[i+1][j]

temp[i+1][j]=-1

return temp

if movement=="left":

for i in range(3):

for j in range(3):

if(temp[i][j]==-1):

if j!=0:

temp[i][j]=temp[i][j-1]

temp[i][j-1]=-1

return temp

if movement=="right":

for i in range(3):

for j in range(3):

if(temp[i][j]==-1):

if j!=2:

temp[i][j]=temp[i][j+1]

temp[i][j+1]=-1

return temp

def ids():

global inp

global out

global flag

for limit in range(100):

print('LIMIT -> '+str(limit))

stack=[]

inpx=[inp,"none"]

stack.append(inpx)

level=0

while(True):

if len(stack)==0:

break

puzzle=stack.pop(0)

if level<=limit:

print(str(puzzle[1])+" --> "+str(puzzle[0]))

if(puzzle[0]==out):

print("Found")

print('Path cost='+str(level))

flag=True

return

else:

level=level+1

if(puzzle[1]!="down"):

temp=copy.deepcopy(puzzle[0])

up=move(temp, "up")

if(up!=puzzle[0]):

upx=[up,"up"]

stack.insert(0, upx)

if(puzzle[1]!="right"):

temp=copy.deepcopy(puzzle[0])

left=move(temp, "left")

if(left!=puzzle[0]):

leftx=[left,"left"]

stack.insert(0, leftx)

if(puzzle[1]!="up"):

temp=copy.deepcopy(puzzle[0])

down=move(temp, "down")

if(down!=puzzle[0]):

downx=[down,"down"]

stack.insert(0, downx)

if(puzzle[1]!="left"):

temp=copy.deepcopy(puzzle[0])

right=move(temp, "right")

if(right!=puzzle[0]):

rightx=[right,"right"]

stack.insert(0, rightx)

print('~~~~~~~~~~~~ IDS ~~~~~~~~~~~~')

ids()

Output:

~~~~~~~~~~~~ IDS ~~~~~~~~~~~~

LIMIT -> 0

none --> [[1, 2, 3], [4, -1, 5], [6, 7, 8]]

LIMIT -> 1

none --> [[1, 2, 3], [4, -1, 5], [6, 7, 8]]

right --> [[1, 2, 3], [4, 5, -1], [6, 7, 8]]

LIMIT -> 2

none --> [[1, 2, 3], [4, -1, 5], [6, 7, 8]]

right --> [[1, 2, 3], [4, 5, -1], [6, 7, 8]]

down --> [[1, 2, 3], [4, 5, 8], [6, 7, -1]]

LIMIT -> 3

none --> [[1, 2, 3], [4, -1, 5], [6, 7, 8]]

right --> [[1, 2, 3], [4, 5, -1], [6, 7, 8]]

down --> [[1, 2, 3], [4, 5, 8], [6, 7, -1]]

left --> [[1, 2, 3], [4, 5, 8], [6, -1, 7]]

LIMIT -> 4

none --> [[1, 2, 3], [4, -1, 5], [6, 7, 8]]

right --> [[1, 2, 3], [4, 5, -1], [6, 7, 8]]

down --> [[1, 2, 3], [4, 5, 8], [6, 7, -1]]

left --> [[1, 2, 3], [4, 5, 8], [6, -1, 7]]

left --> [[1, 2, 3], [4, 5, 8], [-1, 6, 7]]

LIMIT -> 5

none --> [[1, 2, 3], [4, -1, 5], [6, 7, 8]]

right --> [[1, 2, 3], [4, 5, -1], [6, 7, 8]]

down --> [[1, 2, 3], [4, 5, 8], [6, 7, -1]]

left --> [[1, 2, 3], [4, 5, 8], [6, -1, 7]]

left --> [[1, 2, 3], [4, 5, 8], [-1, 6, 7]]

up --> [[1, 2, 3], [-1, 5, 8], [4, 6, 7]]

LIMIT -> 6

none --> [[1, 2, 3], [4, -1, 5], [6, 7, 8]]

right --> [[1, 2, 3], [4, 5, -1], [6, 7, 8]]

down --> [[1, 2, 3], [4, 5, 8], [6, 7, -1]]

left --> [[1, 2, 3], [4, 5, 8], [6, -1, 7]]

left --> [[1, 2, 3], [4, 5, 8], [-1, 6, 7]]

up --> [[1, 2, 3], [-1, 5, 8], [4, 6, 7]]

right --> [[1, 2, 3], [5, -1, 8], [4, 6, 7]]

LIMIT -> 7

none --> [[1, 2, 3], [4, -1, 5], [6, 7, 8]]

right --> [[1, 2, 3], [4, 5, -1], [6, 7, 8]]

down --> [[1, 2, 3], [4, 5, 8], [6, 7, -1]]

left --> [[1, 2, 3], [4, 5, 8], [6, -1, 7]]

left --> [[1, 2, 3], [4, 5, 8], [-1, 6, 7]]

up --> [[1, 2, 3], [-1, 5, 8], [4, 6, 7]]

right --> [[1, 2, 3], [5, -1, 8], [4, 6, 7]]

right --> [[1, 2, 3], [5, 8, -1], [4, 6, 7]]

LIMIT -> 8

none --> [[1, 2, 3], [4, -1, 5], [6, 7, 8]]

right --> [[1, 2, 3], [4, 5, -1], [6, 7, 8]]

down --> [[1, 2, 3], [4, 5, 8], [6, 7, -1]]

left --> [[1, 2, 3], [4, 5, 8], [6, -1, 7]]

left --> [[1, 2, 3], [4, 5, 8], [-1, 6, 7]]

up --> [[1, 2, 3], [-1, 5, 8], [4, 6, 7]]

right --> [[1, 2, 3], [5, -1, 8], [4, 6, 7]]

right --> [[1, 2, 3], [5, 8, -1], [4, 6, 7]]

down --> [[1, 2, 3], [5, 8, 7], [4, 6, -1]]

LIMIT -> 9

none --> [[1, 2, 3], [4, -1, 5], [6, 7, 8]]

right --> [[1, 2, 3], [4, 5, -1], [6, 7, 8]]

down --> [[1, 2, 3], [4, 5, 8], [6, 7, -1]]

left --> [[1, 2, 3], [4, 5, 8], [6, -1, 7]]

left --> [[1, 2, 3], [4, 5, 8], [-1, 6, 7]]

up --> [[1, 2, 3], [-1, 5, 8], [4, 6, 7]]

right --> [[1, 2, 3], [5, -1, 8], [4, 6, 7]]

right --> [[1, 2, 3], [5, 8, -1], [4, 6, 7]]

down --> [[1, 2, 3], [5, 8, 7], [4, 6, -1]]

left --> [[1, 2, 3], [5, 8, 7], [4, -1, 6]]

LIMIT -> 10

none --> [[1, 2, 3], [4, -1, 5], [6, 7, 8]]

right --> [[1, 2, 3], [4, 5, -1], [6, 7, 8]]

down --> [[1, 2, 3], [4, 5, 8], [6, 7, -1]]

left --> [[1, 2, 3], [4, 5, 8], [6, -1, 7]]

left --> [[1, 2, 3], [4, 5, 8], [-1, 6, 7]]

up --> [[1, 2, 3], [-1, 5, 8], [4, 6, 7]]

right --> [[1, 2, 3], [5, -1, 8], [4, 6, 7]]

right --> [[1, 2, 3], [5, 8, -1], [4, 6, 7]]

down --> [[1, 2, 3], [5, 8, 7], [4, 6, -1]]

left --> [[1, 2, 3], [5, 8, 7], [4, -1, 6]]

left --> [[1, 2, 3], [5, 8, 7], [-1, 4, 6]]

LIMIT -> 11

none --> [[1, 2, 3], [4, -1, 5], [6, 7, 8]]

right --> [[1, 2, 3], [4, 5, -1], [6, 7, 8]]

down --> [[1, 2, 3], [4, 5, 8], [6, 7, -1]]

left --> [[1, 2, 3], [4, 5, 8], [6, -1, 7]]

left --> [[1, 2, 3], [4, 5, 8], [-1, 6, 7]]

up --> [[1, 2, 3], [-1, 5, 8], [4, 6, 7]]

right --> [[1, 2, 3], [5, -1, 8], [4, 6, 7]]

right --> [[1, 2, 3], [5, 8, -1], [4, 6, 7]]

down --> [[1, 2, 3], [5, 8, 7], [4, 6, -1]]

left --> [[1, 2, 3], [5, 8, 7], [4, -1, 6]]

left --> [[1, 2, 3], [5, 8, 7], [-1, 4, 6]]

up --> [[1, 2, 3], [-1, 8, 7], [5, 4, 6]]

LIMIT -> 12

none --> [[1, 2, 3], [4, -1, 5], [6, 7, 8]]

right --> [[1, 2, 3], [4, 5, -1], [6, 7, 8]]

down --> [[1, 2, 3], [4, 5, 8], [6, 7, -1]]

left --> [[1, 2, 3], [4, 5, 8], [6, -1, 7]]

left --> [[1, 2, 3], [4, 5, 8], [-1, 6, 7]]

up --> [[1, 2, 3], [-1, 5, 8], [4, 6, 7]]

right --> [[1, 2, 3], [5, -1, 8], [4, 6, 7]]

right --> [[1, 2, 3], [5, 8, -1], [4, 6, 7]]

down --> [[1, 2, 3], [5, 8, 7], [4, 6, -1]]

left --> [[1, 2, 3], [5, 8, 7], [4, -1, 6]]

left --> [[1, 2, 3], [5, 8, 7], [-1, 4, 6]]

up --> [[1, 2, 3], [-1, 8, 7], [5, 4, 6]]

right --> [[1, 2, 3], [8, -1, 7], [5, 4, 6]]

LIMIT -> 13

none --> [[1, 2, 3], [4, -1, 5], [6, 7, 8]]

right --> [[1, 2, 3], [4, 5, -1], [6, 7, 8]]

down --> [[1, 2, 3], [4, 5, 8], [6, 7, -1]]

left --> [[1, 2, 3], [4, 5, 8], [6, -1, 7]]

left --> [[1, 2, 3], [4, 5, 8], [-1, 6, 7]]

up --> [[1, 2, 3], [-1, 5, 8], [4, 6, 7]]

right --> [[1, 2, 3], [5, -1, 8], [4, 6, 7]]

right --> [[1, 2, 3], [5, 8, -1], [4, 6, 7]]

down --> [[1, 2, 3], [5, 8, 7], [4, 6, -1]]

left --> [[1, 2, 3], [5, 8, 7], [4, -1, 6]]

left --> [[1, 2, 3], [5, 8, 7], [-1, 4, 6]]

up --> [[1, 2, 3], [-1, 8, 7], [5, 4, 6]]

right --> [[1, 2, 3], [8, -1, 7], [5, 4, 6]]

right --> [[1, 2, 3], [8, 7, -1], [5, 4, 6]]

LIMIT -> 14

none --> [[1, 2, 3], [4, -1, 5], [6, 7, 8]]

right --> [[1, 2, 3], [4, 5, -1], [6, 7, 8]]

down --> [[1, 2, 3], [4, 5, 8], [6, 7, -1]]

left --> [[1, 2, 3], [4, 5, 8], [6, -1, 7]]

left --> [[1, 2, 3], [4, 5, 8], [-1, 6, 7]]

up --> [[1, 2, 3], [-1, 5, 8], [4, 6, 7]]

right --> [[1, 2, 3], [5, -1, 8], [4, 6, 7]]

right --> [[1, 2, 3], [5, 8, -1], [4, 6, 7]]

down --> [[1, 2, 3], [5, 8, 7], [4, 6, -1]]

left --> [[1, 2, 3], [5, 8, 7], [4, -1, 6]]

left --> [[1, 2, 3], [5, 8, 7], [-1, 4, 6]]

up --> [[1, 2, 3], [-1, 8, 7], [5, 4, 6]]

right --> [[1, 2, 3], [8, -1, 7], [5, 4, 6]]

right --> [[1, 2, 3], [8, 7, -1], [5, 4, 6]]

down --> [[1, 2, 3], [8, 7, 6], [5, 4, -1]]

LIMIT -> 15

none --> [[1, 2, 3], [4, -1, 5], [6, 7, 8]]

right --> [[1, 2, 3], [4, 5, -1], [6, 7, 8]]

down --> [[1, 2, 3], [4, 5, 8], [6, 7, -1]]

left --> [[1, 2, 3], [4, 5, 8], [6, -1, 7]]

left --> [[1, 2, 3], [4, 5, 8], [-1, 6, 7]]

up --> [[1, 2, 3], [-1, 5, 8], [4, 6, 7]]

right --> [[1, 2, 3], [5, -1, 8], [4, 6, 7]]

right --> [[1, 2, 3], [5, 8, -1], [4, 6, 7]]

down --> [[1, 2, 3], [5, 8, 7], [4, 6, -1]]

left --> [[1, 2, 3], [5, 8, 7], [4, -1, 6]]

left --> [[1, 2, 3], [5, 8, 7], [-1, 4, 6]]

up --> [[1, 2, 3], [-1, 8, 7], [5, 4, 6]]

right --> [[1, 2, 3], [8, -1, 7], [5, 4, 6]]

right --> [[1, 2, 3], [8, 7, -1], [5, 4, 6]]

down --> [[1, 2, 3], [8, 7, 6], [5, 4, -1]]

left --> [[1, 2, 3], [8, 7, 6], [5, -1, 4]]

LIMIT -> 16

none --> [[1, 2, 3], [4, -1, 5], [6, 7, 8]]

right --> [[1, 2, 3], [4, 5, -1], [6, 7, 8]]

down --> [[1, 2, 3], [4, 5, 8], [6, 7, -1]]

left --> [[1, 2, 3], [4, 5, 8], [6, -1, 7]]

left --> [[1, 2, 3], [4, 5, 8], [-1, 6, 7]]

up --> [[1, 2, 3], [-1, 5, 8], [4, 6, 7]]

right --> [[1, 2, 3], [5, -1, 8], [4, 6, 7]]

right --> [[1, 2, 3], [5, 8, -1], [4, 6, 7]]

down --> [[1, 2, 3], [5, 8, 7], [4, 6, -1]]

left --> [[1, 2, 3], [5, 8, 7], [4, -1, 6]]

left --> [[1, 2, 3], [5, 8, 7], [-1, 4, 6]]

up --> [[1, 2, 3], [-1, 8, 7], [5, 4, 6]]

right --> [[1, 2, 3], [8, -1, 7], [5, 4, 6]]

right --> [[1, 2, 3], [8, 7, -1], [5, 4, 6]]

down --> [[1, 2, 3], [8, 7, 6], [5, 4, -1]]

left --> [[1, 2, 3], [8, 7, 6], [5, -1, 4]]

left --> [[1, 2, 3], [8, 7, 6], [-1, 5, 4]]

LIMIT -> 17

none --> [[1, 2, 3], [4, -1, 5], [6, 7, 8]]

right --> [[1, 2, 3], [4, 5, -1], [6, 7, 8]]

down --> [[1, 2, 3], [4, 5, 8], [6, 7, -1]]

left --> [[1, 2, 3], [4, 5, 8], [6, -1, 7]]

left --> [[1, 2, 3], [4, 5, 8], [-1, 6, 7]]

up --> [[1, 2, 3], [-1, 5, 8], [4, 6, 7]]

right --> [[1, 2, 3], [5, -1, 8], [4, 6, 7]]

right --> [[1, 2, 3], [5, 8, -1], [4, 6, 7]]

down --> [[1, 2, 3], [5, 8, 7], [4, 6, -1]]

left --> [[1, 2, 3], [5, 8, 7], [4, -1, 6]]

left --> [[1, 2, 3], [5, 8, 7], [-1, 4, 6]]

up --> [[1, 2, 3], [-1, 8, 7], [5, 4, 6]]

right --> [[1, 2, 3], [8, -1, 7], [5, 4, 6]]

right --> [[1, 2, 3], [8, 7, -1], [5, 4, 6]]

down --> [[1, 2, 3], [8, 7, 6], [5, 4, -1]]

left --> [[1, 2, 3], [8, 7, 6], [5, -1, 4]]

left --> [[1, 2, 3], [8, 7, 6], [-1, 5, 4]]

up --> [[1, 2, 3], [-1, 7, 6], [8, 5, 4]]

LIMIT -> 18

none --> [[1, 2, 3], [4, -1, 5], [6, 7, 8]]

right --> [[1, 2, 3], [4, 5, -1], [6, 7, 8]]

down --> [[1, 2, 3], [4, 5, 8], [6, 7, -1]]

left --> [[1, 2, 3], [4, 5, 8], [6, -1, 7]]

left --> [[1, 2, 3], [4, 5, 8], [-1, 6, 7]]

up --> [[1, 2, 3], [-1, 5, 8], [4, 6, 7]]

right --> [[1, 2, 3], [5, -1, 8], [4, 6, 7]]

right --> [[1, 2, 3], [5, 8, -1], [4, 6, 7]]

down --> [[1, 2, 3], [5, 8, 7], [4, 6, -1]]

left --> [[1, 2, 3], [5, 8, 7], [4, -1, 6]]

left --> [[1, 2, 3], [5, 8, 7], [-1, 4, 6]]

up --> [[1, 2, 3], [-1, 8, 7], [5, 4, 6]]

right --> [[1, 2, 3], [8, -1, 7], [5, 4, 6]]

right --> [[1, 2, 3], [8, 7, -1], [5, 4, 6]]

down --> [[1, 2, 3], [8, 7, 6], [5, 4, -1]]

left --> [[1, 2, 3], [8, 7, 6], [5, -1, 4]]

left --> [[1, 2, 3], [8, 7, 6], [-1, 5, 4]]

up --> [[1, 2, 3], [-1, 7, 6], [8, 5, 4]]

right --> [[1, 2, 3], [7, -1, 6], [8, 5, 4]]

LIMIT -> 19

none --> [[1, 2, 3], [4, -1, 5], [6, 7, 8]]

right --> [[1, 2, 3], [4, 5, -1], [6, 7, 8]]

down --> [[1, 2, 3], [4, 5, 8], [6, 7, -1]]

left --> [[1, 2, 3], [4, 5, 8], [6, -1, 7]]

left --> [[1, 2, 3], [4, 5, 8], [-1, 6, 7]]

up --> [[1, 2, 3], [-1, 5, 8], [4, 6, 7]]

right --> [[1, 2, 3], [5, -1, 8], [4, 6, 7]]

right --> [[1, 2, 3], [5, 8, -1], [4, 6, 7]]

down --> [[1, 2, 3], [5, 8, 7], [4, 6, -1]]

left --> [[1, 2, 3], [5, 8, 7], [4, -1, 6]]

left --> [[1, 2, 3], [5, 8, 7], [-1, 4, 6]]

up --> [[1, 2, 3], [-1, 8, 7], [5, 4, 6]]

right --> [[1, 2, 3], [8, -1, 7], [5, 4, 6]]

right --> [[1, 2, 3], [8, 7, -1], [5, 4, 6]]

down --> [[1, 2, 3], [8, 7, 6], [5, 4, -1]]

left --> [[1, 2, 3], [8, 7, 6], [5, -1, 4]]

left --> [[1, 2, 3], [8, 7, 6], [-1, 5, 4]]

up --> [[1, 2, 3], [-1, 7, 6], [8, 5, 4]]

right --> [[1, 2, 3], [7, -1, 6], [8, 5, 4]]

right --> [[1, 2, 3], [7, 6, -1], [8, 5, 4]]

LIMIT -> 20

none --> [[1, 2, 3], [4, -1, 5], [6, 7, 8]]

right --> [[1, 2, 3], [4, 5, -1], [6, 7, 8]]

down --> [[1, 2, 3], [4, 5, 8], [6, 7, -1]]

left --> [[1, 2, 3], [4, 5, 8], [6, -1, 7]]

left --> [[1, 2, 3], [4, 5, 8], [-1, 6, 7]]

up --> [[1, 2, 3], [-1, 5, 8], [4, 6, 7]]

right --> [[1, 2, 3], [5, -1, 8], [4, 6, 7]]

right --> [[1, 2, 3], [5, 8, -1], [4, 6, 7]]

down --> [[1, 2, 3], [5, 8, 7], [4, 6, -1]]

left --> [[1, 2, 3], [5, 8, 7], [4, -1, 6]]

left --> [[1, 2, 3], [5, 8, 7], [-1, 4, 6]]

up --> [[1, 2, 3], [-1, 8, 7], [5, 4, 6]]

right --> [[1, 2, 3], [8, -1, 7], [5, 4, 6]]

right --> [[1, 2, 3], [8, 7, -1], [5, 4, 6]]

down --> [[1, 2, 3], [8, 7, 6], [5, 4, -1]]

left --> [[1, 2, 3], [8, 7, 6], [5, -1, 4]]

left --> [[1, 2, 3], [8, 7, 6], [-1, 5, 4]]

up --> [[1, 2, 3], [-1, 7, 6], [8, 5, 4]]

right --> [[1, 2, 3], [7, -1, 6], [8, 5, 4]]

right --> [[1, 2, 3], [7, 6, -1], [8, 5, 4]]

down --> [[1, 2, 3], [7, 6, 4], [8, 5, -1]]

LIMIT -> 21

none --> [[1, 2, 3], [4, -1, 5], [6, 7, 8]]

right --> [[1, 2, 3], [4, 5, -1], [6, 7, 8]]

down --> [[1, 2, 3], [4, 5, 8], [6, 7, -1]]

left --> [[1, 2, 3], [4, 5, 8], [6, -1, 7]]

left --> [[1, 2, 3], [4, 5, 8], [-1, 6, 7]]

up --> [[1, 2, 3], [-1, 5, 8], [4, 6, 7]]

right --> [[1, 2, 3], [5, -1, 8], [4, 6, 7]]

right --> [[1, 2, 3], [5, 8, -1], [4, 6, 7]]

down --> [[1, 2, 3], [5, 8, 7], [4, 6, -1]]

left --> [[1, 2, 3], [5, 8, 7], [4, -1, 6]]

left --> [[1, 2, 3], [5, 8, 7], [-1, 4, 6]]

up --> [[1, 2, 3], [-1, 8, 7], [5, 4, 6]]

right --> [[1, 2, 3], [8, -1, 7], [5, 4, 6]]

right --> [[1, 2, 3], [8, 7, -1], [5, 4, 6]]

down --> [[1, 2, 3], [8, 7, 6], [5, 4, -1]]

left --> [[1, 2, 3], [8, 7, 6], [5, -1, 4]]

left --> [[1, 2, 3], [8, 7, 6], [-1, 5, 4]]

up --> [[1, 2, 3], [-1, 7, 6], [8, 5, 4]]

right --> [[1, 2, 3], [7, -1, 6], [8, 5, 4]]

right --> [[1, 2, 3], [7, 6, -1], [8, 5, 4]]

down --> [[1, 2, 3], [7, 6, 4], [8, 5, -1]]

left --> [[1, 2, 3], [7, 6, 4], [8, -1, 5]]

LIMIT -> 22

none --> [[1, 2, 3], [4, -1, 5], [6, 7, 8]]

right --> [[1, 2, 3], [4, 5, -1], [6, 7, 8]]

down --> [[1, 2, 3], [4, 5, 8], [6, 7, -1]]

left --> [[1, 2, 3], [4, 5, 8], [6, -1, 7]]

left --> [[1, 2, 3], [4, 5, 8], [-1, 6, 7]]

up --> [[1, 2, 3], [-1, 5, 8], [4, 6, 7]]

right --> [[1, 2, 3], [5, -1, 8], [4, 6, 7]]

right --> [[1, 2, 3], [5, 8, -1], [4, 6, 7]]

down --> [[1, 2, 3], [5, 8, 7], [4, 6, -1]]

left --> [[1, 2, 3], [5, 8, 7], [4, -1, 6]]

left --> [[1, 2, 3], [5, 8, 7], [-1, 4, 6]]

up --> [[1, 2, 3], [-1, 8, 7], [5, 4, 6]]

right --> [[1, 2, 3], [8, -1, 7], [5, 4, 6]]

right --> [[1, 2, 3], [8, 7, -1], [5, 4, 6]]

down --> [[1, 2, 3], [8, 7, 6], [5, 4, -1]]

left --> [[1, 2, 3], [8, 7, 6], [5, -1, 4]]

left --> [[1, 2, 3], [8, 7, 6], [-1, 5, 4]]

up --> [[1, 2, 3], [-1, 7, 6], [8, 5, 4]]

right --> [[1, 2, 3], [7, -1, 6], [8, 5, 4]]

right --> [[1, 2, 3], [7, 6, -1], [8, 5, 4]]

down --> [[1, 2, 3], [7, 6, 4], [8, 5, -1]]

left --> [[1, 2, 3], [7, 6, 4], [8, -1, 5]]

left --> [[1, 2, 3], [7, 6, 4], [-1, 8, 5]]

LIMIT -> 23

none --> [[1, 2, 3], [4, -1, 5], [6, 7, 8]]

right --> [[1, 2, 3], [4, 5, -1], [6, 7, 8]]

down --> [[1, 2, 3], [4, 5, 8], [6, 7, -1]]

left --> [[1, 2, 3], [4, 5, 8], [6, -1, 7]]

left --> [[1, 2, 3], [4, 5, 8], [-1, 6, 7]]

up --> [[1, 2, 3], [-1, 5, 8], [4, 6, 7]]

right --> [[1, 2, 3], [5, -1, 8], [4, 6, 7]]

right --> [[1, 2, 3], [5, 8, -1], [4, 6, 7]]

down --> [[1, 2, 3], [5, 8, 7], [4, 6, -1]]

left --> [[1, 2, 3], [5, 8, 7], [4, -1, 6]]

left --> [[1, 2, 3], [5, 8, 7], [-1, 4, 6]]

up --> [[1, 2, 3], [-1, 8, 7], [5, 4, 6]]

right --> [[1, 2, 3], [8, -1, 7], [5, 4, 6]]

right --> [[1, 2, 3], [8, 7, -1], [5, 4, 6]]

down --> [[1, 2, 3], [8, 7, 6], [5, 4, -1]]

left --> [[1, 2, 3], [8, 7, 6], [5, -1, 4]]

left --> [[1, 2, 3], [8, 7, 6], [-1, 5, 4]]

up --> [[1, 2, 3], [-1, 7, 6], [8, 5, 4]]

right --> [[1, 2, 3], [7, -1, 6], [8, 5, 4]]

right --> [[1, 2, 3], [7, 6, -1], [8, 5, 4]]

down --> [[1, 2, 3], [7, 6, 4], [8, 5, -1]]

left --> [[1, 2, 3], [7, 6, 4], [8, -1, 5]]

left --> [[1, 2, 3], [7, 6, 4], [-1, 8, 5]]

up --> [[1, 2, 3], [-1, 6, 4], [7, 8, 5]]

LIMIT -> 24

none --> [[1, 2, 3], [4, -1, 5], [6, 7, 8]]

right --> [[1, 2, 3], [4, 5, -1], [6, 7, 8]]

down --> [[1, 2, 3], [4, 5, 8], [6, 7, -1]]

left --> [[1, 2, 3], [4, 5, 8], [6, -1, 7]]

left --> [[1, 2, 3], [4, 5, 8], [-1, 6, 7]]

up --> [[1, 2, 3], [-1, 5, 8], [4, 6, 7]]

right --> [[1, 2, 3], [5, -1, 8], [4, 6, 7]]

right --> [[1, 2, 3], [5, 8, -1], [4, 6, 7]]

down --> [[1, 2, 3], [5, 8, 7], [4, 6, -1]]

left --> [[1, 2, 3], [5, 8, 7], [4, -1, 6]]

left --> [[1, 2, 3], [5, 8, 7], [-1, 4, 6]]

up --> [[1, 2, 3], [-1, 8, 7], [5, 4, 6]]

right --> [[1, 2, 3], [8, -1, 7], [5, 4, 6]]

right --> [[1, 2, 3], [8, 7, -1], [5, 4, 6]]

down --> [[1, 2, 3], [8, 7, 6], [5, 4, -1]]

left --> [[1, 2, 3], [8, 7, 6], [5, -1, 4]]

left --> [[1, 2, 3], [8, 7, 6], [-1, 5, 4]]

up --> [[1, 2, 3], [-1, 7, 6], [8, 5, 4]]

right --> [[1, 2, 3], [7, -1, 6], [8, 5, 4]]

right --> [[1, 2, 3], [7, 6, -1], [8, 5, 4]]

down --> [[1, 2, 3], [7, 6, 4], [8, 5, -1]]

left --> [[1, 2, 3], [7, 6, 4], [8, -1, 5]]

left --> [[1, 2, 3], [7, 6, 4], [-1, 8, 5]]

up --> [[1, 2, 3], [-1, 6, 4], [7, 8, 5]]

right --> [[1, 2, 3], [6, -1, 4], [7, 8, 5]]

LIMIT -> 25

none --> [[1, 2, 3], [4, -1, 5], [6, 7, 8]]

right --> [[1, 2, 3], [4, 5, -1], [6, 7, 8]]

down --> [[1, 2, 3], [4, 5, 8], [6, 7, -1]]

left --> [[1, 2, 3], [4, 5, 8], [6, -1, 7]]

left --> [[1, 2, 3], [4, 5, 8], [-1, 6, 7]]

up --> [[1, 2, 3], [-1, 5, 8], [4, 6, 7]]

right --> [[1, 2, 3], [5, -1, 8], [4, 6, 7]]

right --> [[1, 2, 3], [5, 8, -1], [4, 6, 7]]

down --> [[1, 2, 3], [5, 8, 7], [4, 6, -1]]

left --> [[1, 2, 3], [5, 8, 7], [4, -1, 6]]

left --> [[1, 2, 3], [5, 8, 7], [-1, 4, 6]]

up --> [[1, 2, 3], [-1, 8, 7], [5, 4, 6]]

right --> [[1, 2, 3], [8, -1, 7], [5, 4, 6]]

right --> [[1, 2, 3], [8, 7, -1], [5, 4, 6]]

down --> [[1, 2, 3], [8, 7, 6], [5, 4, -1]]

left --> [[1, 2, 3], [8, 7, 6], [5, -1, 4]]

left --> [[1, 2, 3], [8, 7, 6], [-1, 5, 4]]

up --> [[1, 2, 3], [-1, 7, 6], [8, 5, 4]]

right --> [[1, 2, 3], [7, -1, 6], [8, 5, 4]]

right --> [[1, 2, 3], [7, 6, -1], [8, 5, 4]]

down --> [[1, 2, 3], [7, 6, 4], [8, 5, -1]]

left --> [[1, 2, 3], [7, 6, 4], [8, -1, 5]]

left --> [[1, 2, 3], [7, 6, 4], [-1, 8, 5]]

up --> [[1, 2, 3], [-1, 6, 4], [7, 8, 5]]

right --> [[1, 2, 3], [6, -1, 4], [7, 8, 5]]

right --> [[1, 2, 3], [6, 4, -1], [7, 8, 5]]

LIMIT -> 26

none --> [[1, 2, 3], [4, -1, 5], [6, 7, 8]]

right --> [[1, 2, 3], [4, 5, -1], [6, 7, 8]]

down --> [[1, 2, 3], [4, 5, 8], [6, 7, -1]]

left --> [[1, 2, 3], [4, 5, 8], [6, -1, 7]]

left --> [[1, 2, 3], [4, 5, 8], [-1, 6, 7]]

up --> [[1, 2, 3], [-1, 5, 8], [4, 6, 7]]

right --> [[1, 2, 3], [5, -1, 8], [4, 6, 7]]

right --> [[1, 2, 3], [5, 8, -1], [4, 6, 7]]

down --> [[1, 2, 3], [5, 8, 7], [4, 6, -1]]

left --> [[1, 2, 3], [5, 8, 7], [4, -1, 6]]

left --> [[1, 2, 3], [5, 8, 7], [-1, 4, 6]]

up --> [[1, 2, 3], [-1, 8, 7], [5, 4, 6]]

right --> [[1, 2, 3], [8, -1, 7], [5, 4, 6]]

right --> [[1, 2, 3], [8, 7, -1], [5, 4, 6]]

down --> [[1, 2, 3], [8, 7, 6], [5, 4, -1]]

left --> [[1, 2, 3], [8, 7, 6], [5, -1, 4]]

left --> [[1, 2, 3], [8, 7, 6], [-1, 5, 4]]

up --> [[1, 2, 3], [-1, 7, 6], [8, 5, 4]]

right --> [[1, 2, 3], [7, -1, 6], [8, 5, 4]]

right --> [[1, 2, 3], [7, 6, -1], [8, 5, 4]]

down --> [[1, 2, 3], [7, 6, 4], [8, 5, -1]]

left --> [[1, 2, 3], [7, 6, 4], [8, -1, 5]]

left --> [[1, 2, 3], [7, 6, 4], [-1, 8, 5]]

up --> [[1, 2, 3], [-1, 6, 4], [7, 8, 5]]

right --> [[1, 2, 3], [6, -1, 4], [7, 8, 5]]

right --> [[1, 2, 3], [6, 4, -1], [7, 8, 5]]

down --> [[1, 2, 3], [6, 4, 5], [7, 8, -1]]

LIMIT -> 27

none --> [[1, 2, 3], [4, -1, 5], [6, 7, 8]]

right --> [[1, 2, 3], [4, 5, -1], [6, 7, 8]]

down --> [[1, 2, 3], [4, 5, 8], [6, 7, -1]]

left --> [[1, 2, 3], [4, 5, 8], [6, -1, 7]]

left --> [[1, 2, 3], [4, 5, 8], [-1, 6, 7]]

up --> [[1, 2, 3], [-1, 5, 8], [4, 6, 7]]

right --> [[1, 2, 3], [5, -1, 8], [4, 6, 7]]

right --> [[1, 2, 3], [5, 8, -1], [4, 6, 7]]

down --> [[1, 2, 3], [5, 8, 7], [4, 6, -1]]

left --> [[1, 2, 3], [5, 8, 7], [4, -1, 6]]

left --> [[1, 2, 3], [5, 8, 7], [-1, 4, 6]]

up --> [[1, 2, 3], [-1, 8, 7], [5, 4, 6]]

right --> [[1, 2, 3], [8, -1, 7], [5, 4, 6]]

right --> [[1, 2, 3], [8, 7, -1], [5, 4, 6]]

down --> [[1, 2, 3], [8, 7, 6], [5, 4, -1]]

left --> [[1, 2, 3], [8, 7, 6], [5, -1, 4]]

left --> [[1, 2, 3], [8, 7, 6], [-1, 5, 4]]

up --> [[1, 2, 3], [-1, 7, 6], [8, 5, 4]]

right --> [[1, 2, 3], [7, -1, 6], [8, 5, 4]]

right --> [[1, 2, 3], [7, 6, -1], [8, 5, 4]]

down --> [[1, 2, 3], [7, 6, 4], [8, 5, -1]]

left --> [[1, 2, 3], [7, 6, 4], [8, -1, 5]]

left --> [[1, 2, 3], [7, 6, 4], [-1, 8, 5]]

up --> [[1, 2, 3], [-1, 6, 4], [7, 8, 5]]

right --> [[1, 2, 3], [6, -1, 4], [7, 8, 5]]

right --> [[1, 2, 3], [6, 4, -1], [7, 8, 5]]

down --> [[1, 2, 3], [6, 4, 5], [7, 8, -1]]

left --> [[1, 2, 3], [6, 4, 5], [7, -1, 8]]

LIMIT -> 28

none --> [[1, 2, 3], [4, -1, 5], [6, 7, 8]]

right --> [[1, 2, 3], [4, 5, -1], [6, 7, 8]]

down --> [[1, 2, 3], [4, 5, 8], [6, 7, -1]]

left --> [[1, 2, 3], [4, 5, 8], [6, -1, 7]]

left --> [[1, 2, 3], [4, 5, 8], [-1, 6, 7]]

up --> [[1, 2, 3], [-1, 5, 8], [4, 6, 7]]

right --> [[1, 2, 3], [5, -1, 8], [4, 6, 7]]

right --> [[1, 2, 3], [5, 8, -1], [4, 6, 7]]

down --> [[1, 2, 3], [5, 8, 7], [4, 6, -1]]

left --> [[1, 2, 3], [5, 8, 7], [4, -1, 6]]

left --> [[1, 2, 3], [5, 8, 7], [-1, 4, 6]]

up --> [[1, 2, 3], [-1, 8, 7], [5, 4, 6]]

right --> [[1, 2, 3], [8, -1, 7], [5, 4, 6]]

right --> [[1, 2, 3], [8, 7, -1], [5, 4, 6]]

down --> [[1, 2, 3], [8, 7, 6], [5, 4, -1]]

left --> [[1, 2, 3], [8, 7, 6], [5, -1, 4]]

left --> [[1, 2, 3], [8, 7, 6], [-1, 5, 4]]

up --> [[1, 2, 3], [-1, 7, 6], [8, 5, 4]]

right --> [[1, 2, 3], [7, -1, 6], [8, 5, 4]]

right --> [[1, 2, 3], [7, 6, -1], [8, 5, 4]]

down --> [[1, 2, 3], [7, 6, 4], [8, 5, -1]]

left --> [[1, 2, 3], [7, 6, 4], [8, -1, 5]]

left --> [[1, 2, 3], [7, 6, 4], [-1, 8, 5]]

up --> [[1, 2, 3], [-1, 6, 4], [7, 8, 5]]

right --> [[1, 2, 3], [6, -1, 4], [7, 8, 5]]

right --> [[1, 2, 3], [6, 4, -1], [7, 8, 5]]

down --> [[1, 2, 3], [6, 4, 5], [7, 8, -1]]

left --> [[1, 2, 3], [6, 4, 5], [7, -1, 8]]

left --> [[1, 2, 3], [6, 4, 5], [-1, 7, 8]]

Found

Path cost=28

Program 3

Implement 8-puzzle using A\* algorithm

def print\_b(src):

state = src.copy()

state[state.index(-1)] = ' '

print(

f"""

{state[0]} {state[1]} {state[2]}

{state[3]} {state[4]} {state[5]}

{state[6]} {state[7]} {state[8]}

"""

)

def h(state, target):

count=0

i=0

for j in state:

if state[i]!= target[i]:

count=count+1

i=i+1

return count

def astar(state,target):

states = [src]

g = 0

visited\_states =[]

while len(states):

print(f"Level: {g}")

moves = []

for state in states:

visited\_states.append(state)

print\_b(state)

if state == target:

print("Success")

return

moves += [move for move in possible\_moves(state, visited\_states) if move not in moves]

costs = [g + h(move, target) for move in moves]

states = [moves[i] for i in range(len(moves)) if costs[i] == min(costs)]

g += 1

print("Fail")

def possible\_moves(state, visited\_state):

b = state.index(-1);

d = []

if b - 3 in range(9):

d.append('u')

if b not in [0, 3, 6]:

d.append('l')

if b not in [2, 5, 8]:

d.append('r')

if b + 3 in range(9):

d.append('d')

pos\_moves = []

for m in d:

pos\_moves.append(gen(state, m, b))

return [move for move in pos\_moves if move not in visited\_state]

def gen(state, m, b):

temp = state.copy()

if m == 'u': temp[b - 3], temp[b] = temp[b], temp[b - 3]

if m == 'l': temp[b - 1], temp[b] = temp[b], temp[b - 1]

if m == 'r': temp[b + 1], temp[b] = temp[b], temp[b + 1]

if m == 'd': temp[b + 3], temp[b] = temp[b], temp[b + 3]

return temp

src = [1, 2, 3, -1, 4, 6, 7, 5, 8]

target = [1, 2, 3, 4, 5, 6, 7, 8,-1]

astar(src, target)

output:-

Level: 0

1 2 3

4 6

7 5 8

Level: 1

1 2 3

4 6

7 5 8

Level: 2

1 2 3

4 5 6

7 8

Level: 3

1 2 3

4 5 6

7 8

Success